Departamento de Informática

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Desarrollo de una aplicación móvil Android para la gestión de empleo

Autor: Iván Álvarez Arias

Tutor: María Soledad Escolar Díaz

Leganés, junio de 2014

Título: Desarrollo de una aplicación móvil Android para la gestión de empleo

Autor: Iván Álvarez Arias

Director: María Soledad Escolar Díaz

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

A Soledad Escolar, mi tutora en este proyecto. Gracias por el apoyo incondicional incluso en los peores momentos. Le estaré agradecido siempre.

A mis padres, espero que estéis orgullosos del pequeño, el mérito es también vuestro. Gracias por enseñarme a ser lo que soy.

A mi chica Natalia, gracias por sacarme sonrisas cuando más se necesitan. Me siento tremendamente afortunado.

A mis compañeros en la Universidad con los que he trabajado tan duro durante estos años y que serán amigos para toda la vida. Por los proyectos futuros que nos esperan juntos.

Resumen

En este trabajo de fin de grado se desarrolla un sistema distribuido que permite la gestión del empleo, utilizando distintas técnicas y herramientas. El objetivo de esta aplicación es mejorar la situación laboral del usuario que lo utiliza.

La utilización del sistema por parte del usuario se realiza mediante un dispositivo móvil (Smartphone) instalando una aplicación en él. Esta situación permite que el sistema adquiera todos los valores positivos que aportan estos dispositivos como la movilidad o las capacidades técnicas (posicionamiento geográfico, nivel de cómputo, conexión a Internet).

El dispositivo mencionado no es el único elemento del sistema. Una aplicación servidor será la encargada de tratar las peticiones, gestionando la información de los usuarios y aportando las funcionalidades que el sistema requiere. Para realizar estas capacidades, se hará uso de una base de datos que el sistema también desarrolla.

La funcionalidad final del sistema desarrollado permitirá al usuario que lo utilice, una gestión del empleo hasta ahora desconocida gracias al cambio de filosofía del trabajo que pretende llevar a cabo el proyecto.

Abstract

Nowadays, the Spanish economic situation is undergoing a phase of great difficulty. The reason of this critical phase is the unemployment rate that Spain have. According to the current data, more than 4,500,000 people are unemployed and 57.2 percent of young people have not job today.

The main objective of this project is the use of appropriate tools and knowledge to build a system that helps to improve the current situation of the population. The developed system answers a question, what would happen if the work was generated by ordinary people who want to solve a need? With this change of philosophy on how to work, it has been developed the following system.

In this project, different elements are used for their development. The first of these elements is the Android operating system. Android is an operating system designed for mobile devices where direct interaction with the device is done through the touch screen. Currently, we can find android devices in over 190 countries, and according to the official website of android, over 1 million android devices are recorded every day. Also, Android has an app market (Google Play) in which more than 25 billion applications are downloaded per year.

The next item that will use the system is the language of programming *C*. The main feature of this programming language is that it allows to perform similar operations that a programming language for low-level run. This can be used for example for the creation of threads, that can increase the functionality and efficiency of the system that develops. In addition, many libraries implemented allow interaction with programs in other languages.
The storage and management of information is an important point in the system. To do this, the system uses the database manager MySQL. With this element, the system can store information that must be processed, allowing develop the functionality expected.

The project is a distributed system that allows the people who offered a job to contact people who can do it. The system also allows a user to select a particular job. To properly select the future job candidate, geographical position and the knowledge that they have about the field work is used. To do this, a client-server model is implemented. On the client-server model, tasks are divided between the elements that make requests and initiate communications (clients) and the elements that process the requests and prepare a response to clients (server). The system name is *ONE JOB* and it has the following elements:

- The client application is developed on Android, and can be installed on a mobile device. In the same application, you can generate new jobs and accept offers. Also, the system allows to enter the curriculum in the system to be selected in the job generated. The client application will have to make connections to the server to send requests and get the geographical position of the device. For the development of the client application, has been made an intuitive interface, that allows the user to interact with the system easily and effectively. Future users of the application will not have the need of advanced knowledge in the use of technological tools.

- The server system will be implemented in C. The server will get requests from clients; these requests will be processed by the server which will send back a response towards the client. The server is responsible for conducting the selection of candidates when a new job is generated. Server also manages the information that the client application sends. This kind of functions uses a database that stores the information sorted. The server has to make connections to the database to perform the functions.

MySQL will be used as database manager. This manager will store the information in an orderly manner to make efficient queries. The database will be connected to the server system in order to make multiple connections between the system and the database. To

use a database, access control is performed. System security is improved and prevents an attacker to modify information.

For communication between the client and the server must be performed, and for this it is necessary to design a communication protocol. This communication protocol defines the message structure and contents for each service that the system implements. To create an efficient communication protocol will allow the system to improve the speed when performing the functions.

TCP sockets are used for the connection between the system elements. TCP is a connection-oriented protocol that ensures no loss in the delivery, and receipt of information occurs orderly. The other protocol that could be used is UDP. This protocol is not connection oriented, messages losses can be produced in the delivery, and the order of arrival of messages does not have to be in order.

Other aspect to implement es to compute the distances between two users. To select where the user is geographically located, the system uses Google maps. With Google maps, the system retrieves the longitude and latitude where the device is found. Get the longitude and latitude of the device is an important point to make selection of candidates for Jobs.

These are the conclusions of the project once completed:

- **Fulfilled objectives:** Once the project is completed, will be discussed the fulfillment of the objectives discussed. First, the system has to bring together people who generate jobs and people who need work. The system can generate the contact between them with a selection algorithm, using the knowledge of users and their geographical position. The final system allows users to enter their data into the system as follows. Also allows users to generate new jobs. Once the users enter the data, the server takes care of the selection process and all information is stored in the database. Another objective of the system is that user interaction is simple. Therefore it has been implemented an intuitive interface that prevents the user from making errors during execution. All the elements are correctly observed and can be used properly.

- **C.2 Personal evaluation:** Finally, as a personal project evaluation, some great results are obtained. The system can help people who need it, making his life a little bit easier. After the completion of the project, the skills you have acquired during the time at university is surprised. With this knowledge, we can perform as functional systems like the one developed in this project: Making developments that will help our society.

Índice General

Índice de figuras

Índice de tablas

1. Introducción

En la actualidad existen en España según el Ministerio de Empleo y Seguridad Social más de 4500000 personas sin trabajo lo que equivale a tener 1400000 familias sin ingresos. El 57,2 % de los jóvenes menores de 25 años no tienen opción al trabajo. Este proyecto pretende ser un ejemplo de que las nuevas tecnologías y los conocimientos aplicados poseen un gran potencial y pueden aportar algunas soluciones que permitan contrarrestar estos datos económicos.

1.1 Problema y Motivación

En España la necesidad de trabajo es una circunstancia que está afectando enormemente a la calidad de vida de todos los ciudadanos en general, ya que esta situación es determinante para el motor que impulsa nuestra economía. Las empresas cada vez más realizan ajustes en sus plantillas e impiden que nuevos trabajadores se adhieran a ellas.

Pero, ¿qué pasaría si cambiáramos la filosofía del trabajo? ¿Qué provocaría que la gente común que no tiene por qué tener un nivel adquisitivo amplio fuera quién generara los trabajos? ¿Cuál sería la potencialidad de mantener en continuo contacto a gente *normal* con necesidades y personas que puedan solventarlas? Estas y otras preguntas son las que nos hicimos antes de proponer el desarrollo de un sistema como el que se realiza en este proyecto, un sistema distribuido que permite que gente *normal* tenga la habilidad de generar trabajos y que personas con capacidades para solucionarlos, cubran las necesidades obteniendo gracias a ello ganancias económicas.

La potencialidad de una herramienta se basa en las capacidades que adquieren los usuarios cuando la usan y no en la complejidad de las operaciones que realizan. Los dispositivos móviles junto a los sistemas distribuidos permiten conectar a gente de cualquier parte del mundo y en el caso del presente proyecto, a gente que puede ayudarse mutuamente.

El Smartphone se ha convertido en una herramienta imprescindible en nuestra sociedad gracias a las capacidades que han ido adquiriendo estos dispositivos, permitiendo al usuario realizar numerosas operaciones con un nivel elevado de eficiencia. Una implantación de este sistema sería utilizada por un gran número de usuarios potenciales que serían parte fundamental del sistema, ya que un número mayor de usuarios proporciona un mayor número de trabajos generados y trabajadores potenciales para realizarlo.

1.2 Descripción de la solución

En este Trabajo Fin de Grado se pretende desarrollar un sistema que permitirá la gestión de trabajos en los que el generador de los mismos es una persona particular con habilidades específicas y el consumidor del trabajo es una persona que demanda esas mismas necesidades. En el siguiente ejemplo podemos observar un escenario básico de este sistema.

Imaginemos que un usuario, debido a los conocimientos en Física que ha ido adquiriendo durante sus estudios de Ingeniería, se ve con la posibilidad de impartir clases particulares a otros estudiantes que lo necesiten. Este usuario, debería registrarse en el sistema indicando que está dispuesto a dar clases particulares de Física y algunas otras condiciones, por ejemplo, la distancia máxima que está dispuesto a viajar para impartirlas. El sistema por su parte obtiene la posición geográfica del usuario y la información que ha suministrado. Por ejemplo, la Figura 1 muestra cuál es la posición donde se encuentra el usuario y la distancia que está dispuesto a viajar para trabajar.

Figura 1. Posicionamiento de usuario

En ese preciso momento otro usuario podría generar un trabajo en el sistema. En la especificación del trabajo el usuario detalla que el trabajo consiste en ayudar a preparar los exámenes de selectividad y que por lo tanto se necesita a una persona con conocimientos suficientes como para dar clases.

Cuando se realiza la generación del trabajo, el sistema almacena tanto las características de éste (aquello que se pide, en este caso dar clases particulares) como la posición geográfica donde el usuario lo genera. La Figura 2 muestra como la posición geográfica del trabajo generado se encuentra dentro de los límites geográficos que había establecido el primer usuario.

Figura 2. Posicionamiento trabajo

El sistema es capaz de realizar la selección de los candidatos potenciales del trabajo en función del campo de su especialidad (es decir, aquello que pueden hacer) y la posición geográfica de los mismos. En el ejemplo, el sistema pondría en contacto a los dos usuarios ya que el usuario que necesita el trabajo cumple todos los requisitos para convertirse en un candidato potencial del trabajo que el otro usuario ha generado.

1.3 Objetivos

Una vez estudiada la capacidad que se idea implementar, se pueden sentar los objetivos que persigue el proyecto, exponiendo las capacidades que se esperan de él.

En primer lugar se quiere diseñar e implementar un sistema distribuido que permita la gestión del empleo que relacione ofertas de empleo y candidatos. Este sistema debe permitir tanto la adquisición como la generación de distintos tipos de trabajos utilizando para ello una misma aplicación. El usuario podrá hacer uso únicamente de la funcionalidad que él requiera dentro del sistema, es decir, podrá crear nuevos trabajos, obtener trabajos o ambas cosas. Se desea que la interacción con el sistema se realice de la forma más sencilla posible debido a que los futuros usuarios potenciales no tienen por qué tener conocimientos avanzados en el uso de las tecnologías.

Se desea que el sistema a desarrollar sea lo más manejable y movible para el usuario, además de que la herramienta desarrollada llegue al mayor número de personas posibles. Para cumplir este objetivo la aplicación se instalará en un dispositivo móvil gracias a las capacidades que permite éste. El sistema además deberá aprovechar las funcionalidades que ofrecen en la actualidad los dispositivos móviles inteligentes como por ejemplo la posición geográfica.

Otro objetivo es el de implementar una gran variedad de tipos de trabajos que se puedan generar. De este modo, los usuarios potenciales de la aplicación se incrementarían considerablemente y el tráfico de trabajos generados también sería mayor.

Cuando el usuario interactúe con la aplicación podrá realizar de forma intuitiva todas las funcionalidades gracias a la implementación de una interfaz que cumpla los patrones de diseño establecidos.

El diseño de un protocolo de comunicación propio entre los elementos que componen el proyecto, buscando no solo el correcto funcionamiento del mismo sino que además se realice de forma eficiente.

1.4 Estructura del documento

En esta sección se estudiará cual es la estructura que seguirá el presente documento y qué información contiene cada uno de los capítulos que lo compone.
En primer lugar nos encontramos este propio capítulo, en el cual se intenta realizar un enfoque general del proyecto incluyendo las motivaciones encontradas y la solución que el propio proyecto aporta. También se mencionan los objetivos que se persiguen.

Ya en el segundo capítulo nos encontramos el *Estado de la cuestión*, donde se describirán las tecnologías y las herramientas utilizadas, mostrando su historia y aparición y presentando sus características.

El análisis del sistema se realizará en el tercer capítulo, y se aborda qué es el sistema a desarrollar. Se realiza la especificación de requisitos y se establecen los parámetros que debe cumplir el resultado final.

En el cuarto capítulo se abordará el diseño de la aplicación. Se hablará de la arquitectura de la que constará nuestro sistema así como de la funcionalidad que debe realizar cada componente. En este capítulo se comenzarán a realizar los primeros diseños de la interfaz y se establece el protocolo de comunicación entre los diferentes componentes del sistema.

La implementación del sistema se trata en el quinto capítulo donde se podrá encontrar la codificación de las funcionalidades más representativas del sistema para poder entender cómo funcionan realmente.

En el sexto capítulo se muestra la evaluación que se le realiza del sistema. Se podrán ver diferentes pruebas que permitirán conocer cuál es el comportamiento del sistema y qué aspectos del mismo podrían mejorarse como aspecto futuro.

Ya en el último capítulo encontramos las conclusiones que sirven para dar un repaso global a todo lo anteriormente realizado. Se prestará atención a la consecución de los objetivos fijados inicialmente así como a los trabajos futuros que son posibles realizar y el presupuesto del proyecto.

2. Estado de la cuestión

En este capítulo se presentarán las características más importantes de las herramientas y elementos utilizados para el proyecto, exponiendo su evolución y el estado actual de las mismas. Es importante estudiar y analizar la evolución de las tecnologías usadas para comprender el porqué de su uso y las capacidades que nos permiten desempeñar. Todas las tecnologías tendrán un punto fuerte que se tendrá que potenciar y unas deficiencias que se evitarán que afecten lo menos posible al proyecto.

Los dispositivos móviles inteligentes se han convertido en una parte indispensable para todos en la sociedad de la información. Esta situación viene provocada principalmente por el avance, tanto en hardware como en software, de estos dispositivos. Actualmente el Smartphone cuenta con un nivel muy elevado de procesamiento y memoria principal lo que le convierte en un dispositivo que puede aportar grandes funcionalidades de una manera eficiente. Este hardware está acompañado de una multitud de funcionalidades software gracias a las numerosas aplicaciones disponibles, en su mayoría de adquisición gratuita, y que permiten llevar a cabo las acciones que el usuario quiera con la ventaja de la movilidad que permite el dispositivo. También es muy importante el desarrollo en la interacción con el dispositivo que se ha implementado a lo largo de este tiempo, lo que permite que el usuario pueda interactuar con el dispositivo evitando las complicaciones que originaría una pantalla de tamaño reducido o la ausencia de un teclado físico para la introducción de texto.

En primer lugar, entre las tecnologías que vamos estudiar nos encontramos con el sistema operativo del cliente, el cual será Android y se ejecutará sobre un dispositivo móvil (Smartphone).

2.1 Android

Android es un sistema operativo orientado a dispositivos móviles como tabletas y Smartphones que permite interactuar con las funcionalidades de las que estas herramientas disponen de una manera eficiente y sencilla para el usuario, siendo los conocimientos técnicos de éste no necesariamente altos. Está orientado principalmente su uso con dispositivos de pantalla táctil, con la implementación de interfaz característica que eso conlleva. Además de esto, nos encontramos con un mercado de aplicaciones propio, que permite obtener continuas funcionalidades para el usuario.

2.1.1 Historia

El sistema Android surge en el año 2003 [1] de la mano de Andy Rubin, Rich Miner, Nick Sears y Chris White, componentes y fundadores de la empresa Android Inc. La idea original es que este nuevo sistema tuviera un elemento diferencial al resto como es el *context awareness* o conciencia del contexto, es decir, que el dispositivo fuera conocedor del entorno del usuario, a través de información como la ubicación del usuario, la hora del día, etc. para adaptar mejor su comportamiento a estas circunstancias. Inicialmente, el uso de Android estuvo orientado a cámaras digitales aunque pronto comenzaron los desarrollos para los dispositivos móviles inteligentes o *smartphones* debido al gran despliegue que estaban logrando en la sociedad. Con este sistema por ejemplo el dispositivo era conocedor de la hora del día pudiendo encender o no el flash según esta información.

El rumbo de la empresa Android Inc. cambia totalmente en el año 2005 cuando es adquirida por Google. Esta circunstancia permitiría el crecimiento exponencial de los proyectos y objetivos originales de Android, ya que ahora se contaba con el respaldo y los recursos de una gran empresa como ya era Google en aquella época.
Fue en el año 2007 cuando Google y Android tuvieron que realizar un cambio importante en su proyecto de creación del sistema operativo y es que el lanzamiento del nuevo iPhone por parte de la empresa Apple cambió el concepto

de dispositivo móvil incorporando una pantalla totalmente táctil y eliminando el teclado físico usado hasta ahora por todos los terminales. Android tuvo que adaptar su idea inicial a esta nueva tecnología.

Después de todo el desarrollo realizado, el 22 de octubre del 2008 aparece el primer dispositivo con el sistema operativo Android (Android 1.0) el cual contaba con elementos innovadores en relación al resto de sistemas operativos de la época. El menú desplegable, la posibilidad de añadir widgets, la integración con las aplicaciones de google y la capacidad de acceder a un mercado amplio y en continuo crecimiento de aplicaciones fueron unas de estas innovaciones.

Figura 3. Logotipo de Android

2.1.2 Arquitectura de Android

La arquitectura del sistema operativo Android [2] se puede observar en la Figura 4.

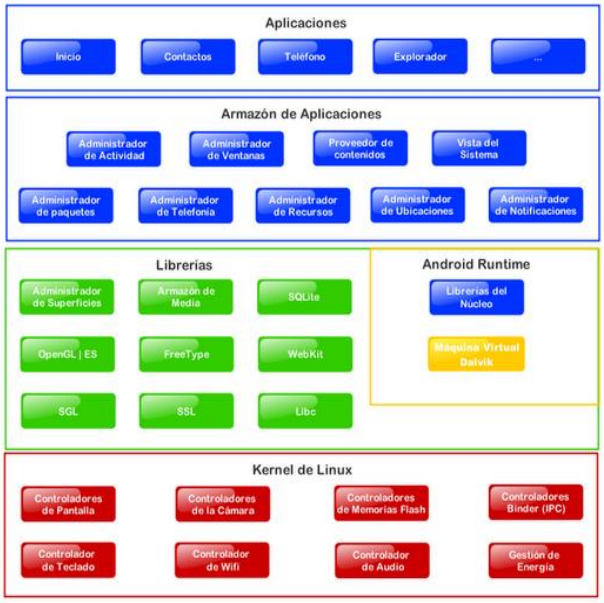

Figura 4. Arquitectura de Android

Se trata de una arquitectura dividida en 4 capas de software, donde el mayor nivel de abstracción corresponde al nivel superior y el menor nivel de abstracción corresponde al nivel inferior.

En el nivel superior podemos observar las aplicaciones. En este nivel se sitúan todas las aplicaciones del usuario incluidas las que trae el dispositivo por defecto y las que el usuario descarga independientemente de su procedencia. Estas aplicaciones obtendrán los recursos para su ejecución de los niveles inferiores a esta "capa" de aplicación.

La siguiente capa es el Framework de las aplicaciones (en la Figura 4 se muestra como Armazón de aplicaciones) observamos todos los recursos que son necesarios para el desarrollo de las aplicaciones. Contiene aquellas APIs[3] que permiten la mayor parte de las funcionalidades de las aplicaciones, permitiendo a los desarrolladores elaborar un gran número variado de funcionalidades.

La siguiente capa se compone de diferentes librerías, gracias a las cuales Android obtiene los recursos necesarios para poder desempeñar sus funcionalidades como sistema operativo. Estas librerías están en lenguaje C y C++. Encontramos por ejemplo librerías que incluyen todas las cabeceras que serán utilizadas (libc), librerías que permiten posicionar correctamente todos los elementos de la pantalla (Surface Manager), que permiten hacer uso del protocolo SSL (Secure Sockets Layer) para conexiones seguras, librerías de codecs para que el dispositivo pueda reproducir distintos tipo de archivos multimedia (Media Libraries), la librería SQLite que permite el uso eficiente de modelos relacionales de bases de datos y una librería que permite la ejecución de un navegador web en el dispositivo (WebKit).

En el mismo nivel que las librerías que hemos visto anteriormente nos encontramos con el entorno de ejecución (Android Runtime). En este caso nos encontramos unas librerías denominadas "Core Libraries" que contienen

numerosas clases de Java que permiten la ejecución de las aplicaciones. Esta ejecución de las aplicaciones sería imposible sin la existencia en este mismo nivel de la máquina virtual Dalvik[4]. Dalvik permite ejecutar los programas Java con un gran rendimiento y sin un consumo excesivo de energía, que son las características más importantes para un dispositivo móvil.

En la última capa se sitúa el kernel de Linux, que utiliza la versión del núcleo de Linux 2.6. En este nivel se realiza una abstracción que permita ejecutar el hardware disponible para el dispositivo. En esta capa encontraremos todos los driver necesarios para el uso de los recursos de hardware del dispositivo.

2.1.3 Dispositivos soportados por Android

Dentro de todos los dispositivos que permiten Android como sistema operativo nos encontramos distintos tipos, cada uno con unas características distintas que aportan diferentes valores al usuario y que permiten que algunas operaciones se realicen mejor o peor en función de las características del dispositivo. Los diferentes dispositivos son los siguientes.

2.1.3.1 Smartphone

El *Smartphone* se caracteriza normalmente por su tamaño reducido, permitiendo una mayor movilidad del mismo. Contiene todas las funcionalidades típicas de los dispositivos que contienen Android tales como conexión a Internet, servicios de ubicación, capacidad de instalar software como aplicaciones, etc. y todo ello con un nivel de eficiencia muy elevado gracias a los desarrollos de dispositivos que se están elaborando continuamente. La evolución de estas herramientas tanto en el plano interno a nivel de software y hardware como a nivel externo en temas de diseño se ha incrementado en los últimos años gracias al gran número de usuarios potenciales que contiene este mercado, situación que las diferentes empresas han sabido aprovechar.

Figura 5. Smartphone Android

2.1.3.2 Tablets

La tablet, o simplemente tableta, es un dispositivo muy similar al Smartphone con la salvedad del tamaño de pantalla. Esta característica es importante en cuanto a la interacción que el usuario realiza con el dispositivo ya que un mayor tamaño de pantalla permite acceder de manera más sencilla a los elementos que ésta contiene. Sin embargo esta situación provoca una pérdida en la movilidad de la herramienta no siendo su transporte tan sencillo como en el caso del Smartphone.

Figura 6. Tableta Android

Además de esto, la tablet a diferencia de lo que ocurre normalmente con el Smartphone no tiene por qué tener asociada una línea telefónica con una tarjeta

SIM, es decir que el dispositivo no está orientado a hacer llamadas aunque si está disponible esta opción en el caso de que el usuario asociara una línea de teléfono a al dispositivo. Se permiten las mismas capacidades que se realizaban con el Smartphone (conexión a Internet, servicios de ubicación, posibilidad de instalar aplicaciones, etc.).

2.1.3.3 Google TV

Con Google TV se desarrolla una televisión totalmente interactiva con el usuario gracias a la incorporación del sistema operativo Android, que le permite instalar numerosas aplicaciones desarrolladas con el fin de emitir distinto tipo de contenido a través de la televisión. Además de esto se permite la integración de otros dispositivos que permiten aumentar las funcionalidades pudiendo también navegar por Internet, realizar video llamadas, etc.

Figura 7. Google TV

2.1.3.4 Portátiles y netbooks

Cada vez más se hace uso del sistema operativo Android para los portátiles y netbooks de tal forma que puedan utilizarse en ellos las aplicaciones desarrolladas para estos dispositivos. Hay que tener en cuenta, que la forma de interactuar con la interfaz en este caso varía ya que se tienen por ejemplo elementos físicos como el teclado y el ratón para la interacción.

Figura 8. Portátil con Android

En la mayoría de las ocasiones, estos dispositivos tienen varios sistemas operativos de tal forma que se pueda dividir las funcionalidades que el usuario requiera entre los distintos sistemas operativos.

2.1.4 Estado actual

Actualmente el smartphone se ha convertido en un elemento indispensable para nuestra sociedad. Esta circunstancia ha sido provocada por las grandes capacidades y funcionalidades que nos permiten este tipo de dispositivos. Android se sitúa como el mayor sistema operativo utilizado en smartphones y por lo tanto uno de los precursores y causantes de la situación actual del teléfono inteligente en nuestra sociedad.

El sistema operativo Android se encuentra disponible actualmente en más de 190 países de todo el mundo como muestra su página oficial para desarrolladores (developer.android.com) registrando más de un millón de nuevos dispositivos Android cada día (se estiman un total de 400 millones de activaciones de dispositivos hasta la fecha), dato muy por encima de sus competidores. La siguiente Figura 9 presenta la comparación de Android con otros sistemas operativos en términos de ventas:

Sistema Operativo	Ventas 2013 (1T)	Share 2013 (1T)	Ventas 2012 (1T)	Share 2012 (1T)
Android	156 millones	74,4 %	83,6 millones	56,9 %
iOS	38 millones	18,2 %	33 millones	22,5%
BlackBerry	6 millones	3 %	9,9 millones	6,8 %
Windows	5,9 millones	2,9 %	2,7 millones	1,9%

Figura 9. Datos de ventas por S.O

Además de las ventas de dispositivos hay que tener en cuenta el mercado de aplicaciones que Android contiene. La Figura 10 presenta el número de descargas de aplicaciones basadas en Android desde el año 2008 hasta el año 2012. Como puede observarse, al finalizar el año 2012 se produjeron 1.5 billones de descargas de aplicaciones al mes y un total de 25 billones de descargas en el año. La tendencia es que este número siga creciendo en los próximos años.

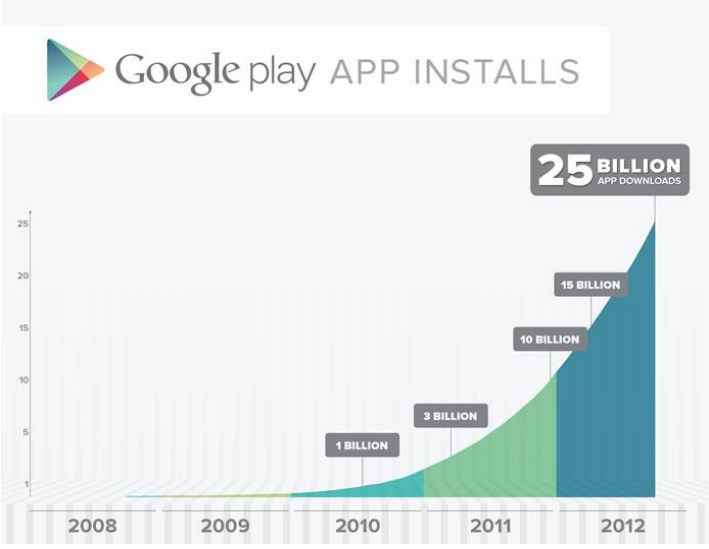

Figura 10. Aplicaciones Android descargadas por año.

Actualmente, Android tiene como versión más actual la 4.4 la cual se encuentra totalmente adaptada para aprovechar al máximo las capacidades de los elementos hardware que contienen actualmente los dispositivos. Respecto a las versiones iniciales de Android, la versión actual permite el uso de nuevas tecnologías como pudo ser en su momento el GPS, los diferentes sensores y la tecnología NFC.

2.1.5 Otros sistemas operativos para dispositivos móviles

Además de Android, encontramos distintos sistemas operativos con características diferenciables y que aportan valores que pueden ser positivos para el usuario que hace uso del dispositivo. Las alternativas a Android más utilizadas son las siguientes.

2.1.5.1 IOS

El sistema operativo IOS está desarrollado por Apple [4] y aunque inicialmente se desarrolló únicamente para iPhone, actualmente forma el sistema operativo de otros dispositivos como Ipod, Ipad o Apple TV.

Al igual que con Android, la interacción con el dispositivo se realiza en la mayor parte gracias a la manipulación directa a través de la pantalla táctil. Las diferencias en IOS residen principalmente en el sistema del mercado de aplicaciones y la actualización del software.

En primer lugar, la filosofía del mercado de aplicaciones desarrolla lo denominado "Walled Garden" que consiste en que Apple tiene total control sobre las aplicaciones que se ofrecen en el mercado de aplicaciones (Apple Store) realizando pruebas para comprobar que cumplen los requisitos que Apple cree conveniente y autenticando en todo momento al desarrollador de la aplicación mediante certificados.

En cuanto a la actualización del sistema operativo, se realiza directamente por parte de Apple independientemente de la operadora. Para utilizar las herramientas de Apple se deberá tener el dispositivo actualizado a la última versión disponible.

2.1.5.2 BlackBerry OS

Este sistema operativo es el usado por los dispositivos BlackBerry [6] y su característica principal es la integración de distintos métodos de entrada de

información por parte del usuario para el dispositivo. De esta forma se tiene elementos como *trackwheel*, *trackball*, *touchpad* y pantallas táctiles que deben ser gestionadas por el sistema operativo.

Este sistema operativo permite además una alta eficiencia en términos de sincronización de correo electrónico y navegación por Internet. Además de esto, tiene un servidor propio para gestionar la sincronización de contactos, tareas y otros elementos del dispositivo.

2.1.5.3 Windows Phone

Windows [7] pone es un sistema operativo para dispositivos móviles desarrollado por Microsoft. Su diferencia principal es la inclusión de una interfaz totalmente diferente (Interfaz Modern) donde los elementos se ordenan por agrupaciones y mosaicos modificables.

Una de las potencialidades de este sistema operativo es la integración con muchas de las herramientas de Microsoft tales como Office o Internet Explorer. En cuanto al mercado de aplicaciones está todavía en auge ya que en comparación con Android o IOS el número de aplicaciones disponibles es mucho menor. Según datos de Microsoft, 15000 nuevas aplicaciones son insertadas en el mercado de aplicaciones cada mes.

2.1.6 Influencia de sus características en el proyecto

En primer lugar Android permite la comunicación a través de Internet. Esta funcionalidad estará presente en el framework de los recursos que hemos visto anteriormente y en las diferentes librerías. Esta posibilidad de comunicarse con Internet permitirá el intercambio de mensajes con el mismo y otro tipo de dispositivos, como pueden ser PCs o portátiles.

Utilizando las APIs de las que disponemos en la arquitectura antes expuesta de Android podemos elaborar mensajes que sean válidos para distintos lenguajes de programación. Esta circunstancia nos permite que el dispositivo con el que se

realiza el intercambio de datos, por ejemplo un servidor de datos, se encuentre en un lenguaje distinto al del cliente y que el funcionamiento se realice correctamente. En cuanto a los usuarios potenciales que tiene la aplicación podemos establecer un número muy elevado teniendo en cuenta los datos mostrados anteriormente en la Figura 10. Con el número de descargas que se realizan cada día la aplicación podrá tener una número potencialmente muy alto de usuarios que hagan la aplicación más efectiva ya que hay que tener en cuenta que si el número de usuarios que la usan es muy elevado, también se incrementará el valor del proyecto porque se encontrarán más trabajos y más trabajadores.

En conclusión podemos establecer que el gran número de usuarios que pueden utilizar la aplicación si ésta está desarrollada en Android es una característica muy importante que afecta directamente a su elección como sistema operativo para el cliente.

2.2 Lenguaje C

El lenguaje de programación C [8] contiene numerosas características que aportan grandes capacidades al desarrollador. Para estudiar este lenguaje y sus características principales expondremos la historia de su creación y las características principales del propio lenguaje, analizando las potencialidades que contiene.

2.2.1 Historia

El lenguaje de programación C se desarrolló en el año 1972 en los laboratorios Bell por Dennis M. Ritchie siendo el resultado de la evolución del anterior lenguaje denominado 'B' [9]. El lenguaje en C surgió como alternativa para la recodificación del sistema Unix, el cual hasta el momento solo era posible realizar mediante ensamblador [10].

El lenguaje de ensamblador era característico de cada máquina (cada arquitectura tiene un juego de instrucciones) lo que dificultaba la unificación. Esta circunstancia

era solventada con la incorporación del lenguaje B el cual permitía la portabilidad de código de una máquina a otra sin la necesidad de estudiar el juego de instrucciones de la propia máquina.

Tomando como base el lenguaje B, se añadieron estructuras de datos que mejoraron la eficiencia del lenguaje así como la incorporación de numerosos operadores y tipos de datos que permitían la programación tanto en bajo como en alto nivel. Este lenguaje de programación resultante es el lenguaje C.

En los años 80 se convierte en un lenguaje muy usado por los desarrolladores lo que permite sustituir a Basic en muchas ocasiones. Además de esto tuvo un gran impulso con la utilización en los IBM PC.

Fue tan grande su expansión y utilización que en el año 1983 el Instituto Nacional Estadounidense de Estándares (ANSI) realizó el C89 como estándar para el lenguaje. En este periodo fue cuando no se realizaron cambios en lenguaje C y las innovaciones que se fueron sucediendo fueron incorporadas a un lenguaje más evolucionado como sería C++. En la actualidad el lenguaje C es utilizado para software de sistema y para el desarrollo de distintas aplicaciones.

2.2.2 Características

El lenguaje C tiene unas características propias que le permiten realizar algunas acciones de manera mucho más eficiente que el resto de lenguajes de programación. Algunas de estas características son las siguientes.

En primer lugar nos encontramos en el núcleo del lenguaje de programación una serie de funciones matemáticas y trato con archivos mediante bibliotecas que permiten realizar las funcionalidades que se requieran tanto a bajo como alto nivel. Además de esto, la programación para el desarrollador no es una tarea complicada gracias a la opción de poder hacerlo de manera estructurada y no estructurada. También se añadieron evoluciones del lenguaje B, un ejemplo de esto es la modificación de algunas operaciones que no tenían sentido realizar pero que sin

embargo se calculaban, provocando un aumento importante de tiempo de ejecución en los casos en los que estas operaciones aparecían.

Una de las características más importantes de C y que le diferenciaron del resto de lenguajes es la dualidad que permitía entre el lenguaje de alto y bajo nivel. Esta circunstancia provoca por lo tanto tener acceso a memoria de bajo nivel con la aparición de los punteros, pudiendo realizar numerosas acciones que en otro lenguaje con esta funcionalidad sería mucho más complicado.

Otra funcionalidad que se observa en lenguaje C es el paso de argumentos de las funciones por valor. Mediante este mecanismo el argumento de la función no era un puntero a una dirección de memoria sino que era una variable que había sido copiada y que su modificación no afectaba en absoluto al elemento original. Esta funcionalidad acompañada de la utilización de acciones de bajo nivel se convirtió en una funcionalidad muy interesante.

Además de esto, aparecen unos tipos de datos agregados no presentes en el lenguaje B, las estructuras (*struct*) que permitían encapsular variables de distinto tipo y ser tratadas como una sola. Este elemento es muy interesante para el desarrollador cuando aparecen datos agregados de forma muy repetida que son difícilmente tratables de manera individual.

2.2.3 Influencia de las características en el proyecto

El lenguaje de programación utilizado para el desarrollo del servidor del sistema será el lenguaje C, permitiendo llevar a cabo distintas funcionalidades que en otro lenguaje de programación sería imposible o con un rendimiento menor. La principal característica por la que se utiliza este lenguaje de programación para implementar el servidor es su capacidad para realizar acciones de bajo nivel, sin perder la facilidad para el desarrollador de llevarlas a cabo.

Es fundamental por tanto a la hora de crear procesos ligeros que permitan llevar a cabo una concurrencia en el servidor. Otros lenguajes permiten también la creación de estos procesos pero no con la eficiencia y la robustez que permiten las características de un lenguaje a bajo nivel. Este lenguaje por tanto permitirá generar hilos que sean los encargados de tratar las peticiones de los clientes pudiendo ser estas simultáneas, lo que es una cualidad esencial si se quiere un servidor que cumpla con los requisitos que debe tener el sistema.

2.3 MySQL

El almacenamiento de la información del sistema de forma sencilla, eficiente y segura es un punto muy importante en cualquier desarrollo. A continuación se analizarán las características principales de MySQL para poder así estudiar los valores positivos que tendría su incorporación en un sistema.

2.3.1 Historia

MySQL[11] aparece en los años 80 de la mano de Michael Widenius, el cual buscaba una forma de almacenar la información que fuera satisfactoria para sus necesidades y objetivos. Este producto inicial comienza a evolucionar en el año 1995 cuando gracias a la colaboración de David Axmark se incluyen nuevos elementos que mejoran enormemente las capacidades de la herramienta [12].

En primer lugar se permite la utilización del lenguaje SQL para interactuar de forma eficiente con todos los elementos disponibles, pudiendo facilitar la tarea del desarrollador enormemente. Así mismo se habilitó el acceso y comunicación con internet lo que se convirtió en una novedad para la época en la que se situaba. De esta forma se originaba MySQL y una empresa encargada de su gestión MySQL AB.
El posterior crecimiento de MySQL viene dado gracias a las numerosas aportaciones de desarrolladores de todo el mundo, que aprovechando el código abierto de la herramienta, consiguen añadir mejoras, que si son lo suficientemente buenas, aparecerían en el producto final.

En Octubre del año 2005 Oracle se hace con la empresa innobase OY que realizó los desarrollos en los que se basa MySQL para realizar transacciones y utilizar la clave foránea. Fue en Enero del 2008 cuando la empresa MySQL es comprada por Sun Microsystems por un total de mil millones de euros. Finalmente en el año 2009 Oracle se hace con la empresa Sun Microsystems y en consecuencia con MySQL.

2.3.2 Características

Las características principales de MySQL son las siguientes:

- Está escrito en C y C++, pudiendo ser ejecutable en un gran número de plataformas. Además de esto se han realizado pruebas con numerosos compiladores arrojando en todos los casos grandes resultados de eficiencia.
- Se encuentra actualmente muy optimizado en términos de multiprocesamiento con la posibilidad de dividir sus operaciones en múltiples hilos que permiten la ejecución en paralelo.
- Se aporta gran seguridad ya que el usuario debe autenticarse siempre ante el sistema para conocer y ejecutar sus posibles privilegios. Toda la información que se envía para la identificación se encuentra cifrada al realizar la conexión con el servidor.
- Los clientes pueden realizar conexiones mediante sockets y otros mecanismos de manera eficiente y segura para obtener la información requerida.

2.3.3 Influencia de las características en el proyecto

La elección de MySQL como herramienta para almacenar la información del sistema se basa en sus distintas características que aportan valores positivos al proyecto. La primera de estas características es la facilidad en la adquisición y posterior instalación en el servidor. Esto se debe principalmente, como se ha expuesto anteriormente, a la existencia de código abierto y la filosofía de seguir creciendo gracias a la ayuda de todos los desarrolladores.

El hecho de que la herramienta sea interoperable ha permitido la inclusión en el lenguaje del servidor de manera sencilla tanto para el lenguaje utilizado en el servidor como para el sistema operativo en el que se ejecuta. En este punto, el gran número de APIs existentes facilita el trabajo al desarrollador y le permiten obtener el máximo rendimiento de la herramienta.

En cuanto a la eficiencia de las acciones que se precisan, MySQL cumple totalmente con los requisitos necesarios ya que permite la ejecución de las tareas de forma robusta, segura y rápida. En capítulos posteriores se analizarán tiempos de ejecución de las distintas tareas obteniendo unos resultados suficientemente buenos como para descartar utilizar otra herramienta similar.

2.4 Mercado de Aplicaciones

En el mercado actual de aplicaciones diseñadas para dispositivos móviles nos encontramos con una serie de aplicaciones que permiten realizar objetivos similares a los que obtenemos con el proyecto. Este apartado pretende presentar algunas de las aplicaciones ya existentes que tienen objetivos comunes con nuestra aplicación y compararlas con nuestra aplicación. La mayoría de estas aplicaciones se basan en datos de otras páginas (InfoJobs, Infoempleo, Computrabajo), siendo estas ofertas demasiado masificadas (el número de candidatos al trabajo es muy alto). Los criterios de búsqueda de estas aplicaciones suelen estar definidos mediante capacidades del futuro empleado y disponibilidad geográfica. Algunas de estas aplicaciones son las siguientes.

- Laboris empleo: En esta aplicación la persona que busca el trabajo tiene que realizar la búsqueda mediante filtros, para poder llegar a las ofertas que realmente le interesan. Además de esto se permite llevar un seguimiento de la oferta a la que uno se subscribe para obtener información sobre el proceso de elección. Los trabajos disponibles son los mismos que los que encontramos en la página web de Laboris[13].

Figura 11. Aplicación Laboris Empleo

- Cb Trabajo y empleo: En este caso la aplicación obtiene tu posición geográfica para filtrar aquellas ofertas que pueden resultar más interesantes. También se permiten distintas formas de introducir tu propio currículum en el sistema y establecer las condiciones de envío de este currículum a las empresas. Los trabajos disponibles también pueden verse a través de la página web de careerBuider[14].

Figura 12. Aplicación CareerBuilder

- Trovit Empleo: En este caso las búsquedas de las ofertas se realizan mediante palabras clave. De esta forma se pueden enviar alertas que permitan asociar correctamente a las personas y los empleos que cumplen con las características necesarias gracias a que contienen esas palabras clave. Los trabajos y su gestión también puede realizarse a través de la página web de Trovit [15].

Figura 13. Aplicación Trovit Empleo

- Uber: Esta aplicación se utiliza para la gestión de viajes ya que mediante la localización del dispositivo utilizando el GPS del mismo, conecta al usuario con el conductor más cercano a él pudiendo recogerle en el lugar donde se encuentra. El registro en el sistema también puede realizarse a de la página web de Uber [16].

Figura 14. Aplicación Uber

2.5 Sockets

El socket es un mecanismo por el cual se permite la comunicación entre dos procesos, posibilitando el intercambio de datos entre computadoras heterogéneas. Esta comunicación se puede realizar cuando los programas conocen las direcciones IP de las máquinas con las que tienen que comunicarse y el número de puerto que identifica al otro proceso.

El socket permite realizar conexiones en muchas ocasiones cuando el modelo utilizado es el de cliente-servidor. El programa que inicia la comunicación es el llamado cliente que normalmente envía peticiones al servidor. El servidor por su parte se encarga de procesar las peticiones que le llegan de las aplicaciones clientes y enviar respuestas a los mismos

2.5.1 Aparición de los Sockets

Cuando se comenzaron a implementar los primeros desarrollos utilizando Internet, eran necesarios implementar protocolos que permitieran obtener toda la potencialidad que contenía la red. Los primeros ordenadores en realizar estos primeros desarrollos fueron los de la Universidad de Berkeley que terminaron desarrollando un mecanismo para comunicar dos programas de manera sencilla y eficaz que denominaron Socket.

Su implantación en los programas que requerían de una comunicación otros programas se extendió rápidamente hasta el punto de ser el principal método utilizado cuando la arquitectura del sistema distribuido era de cliente-servidor.

Actualmente existen multitud de bibliotecas ya implementadas para distintos lenguajes de programación que permiten a los desarrolladores la utilización de Sockets de manera sencilla.

2.5.2 Características de los Sockets

Los sockets permiten utilizar distintos protocolos de transporte para la implementación de aplicaciones. El protocolo en el que encontramos mayor número de implementaciones es Transmission Control Protocol (TCP) [17], aunque también puede usarse el protocolo User Datagram Protocol (UDP) [18].

Los sockets que utilizan el protocolo TCP son orientados a conexión. TCP cuenta entre sus características la fiabilidad, lo que implica que una aplicación que usa sockets basados en TCP puede tener la garantía de que la información que se envía entre sockets llegará al destino sin errores y en el orden correcto. Esta implementación es recomendable para aquellos programas que envíen información cuya pérdida u orden de llegada tenga una importancia clave para su funcionalidad.

En el caso de que el socket este asociado al protocolo de transporte UDP no se tendrá asegurado que la información que se envía llega al destinatario. Además de

esto cuando la información llega al otro proceso no se puede garantizar tampoco que la recepción se realiza en el mismo orden que el envío.

3. Análisis

En este capítulo se pretende analizar el sistema a desarrollar, estudiando aspectos como las características generales de éste, sus particularidades y qué requisitos se deben cumplir. Todos los aspectos que se muestran a continuación condicionarán el proceso de desarrollo, y por lo tanto se realizará un análisis en profundidad que permita posteriormente realizar un diseño acorde y efectivo.

3.1 Propósito

Se pretende desarrollar un sistema distribuido que permita poner en contacto a gente que ofrece trabajo y gente que lo necesita. Los trabajos que se pueden ofertar tienen la peculiaridad de que son ofertados por personas individuales o particulares que buscan cubrir una necesidad. Por ejemplo, una de estas necesidades puede ser solucionar un problema puntual con un ordenador que tenemos en casa y que se ha estropeado. El sistema deberá poner en contacto a la persona que tiene el problema (su ordenador ha dejado de funcionar) con la persona que posee la solución y pueda cubrir la necesidad (persona con estudios en Informática). Para llevar a cabo la asociación entre estas dos personas se prestará atención a características como los conocimientos del candidato a cubrir la necesidad y su localización geográfica. La Figura 15 representa la interacción de los usuarios con el sistema.

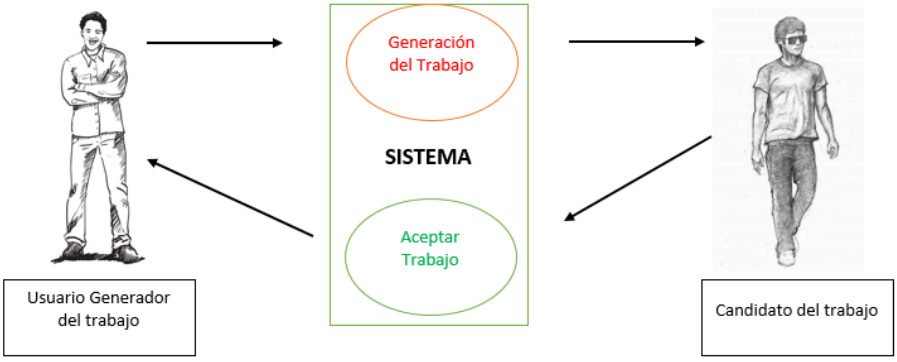

Figura 15. Representación de interacción

Es importante resaltar que en nuestra aplicación los trabajos ofertados/demandados no son los típicos ofrecidos por una empresa con contratos de larga duración. La filosofía del sistema es generar una cantidad muy alta de trabajos que cubran necesidades individuales, siendo un complemento perfecto para aquellos trabajadores que quieran obtener ganancias mientras no encuentran un trabajo estable o que quieran beneficios secundarios a las que ya les aporta su trabajo principal.

Esta herramienta no es solo utilizada por trabajadores individuales sino que pequeñas y medianas empresas también pueden hacer uso de ella para captar nuevos trabajos, por ejemplo pintores, fontaneros o rehabilitaciones en general. De este modo una persona que quiera pintar su casa solo deberá generar el trabajo en el sistema para obtener la lista de pintores que han aceptado el trabajo y que pueden cubrir la necesidad.

Un aspecto importante sobre el sistema es que su cometido es poner en contacto al ofertante de un trabajo con el demandante, el futuro contrato que pueda existir entre ambos no es capacidad del sistema.

3.2 Ámbito del sistema

En esta sección se abordarán temas como el nombre que recibirá el sistema, los objetivos y los valores positivos que se aportan al usuario que los pueda utilizar así como las características de éste.

3.2.1 Nombre del sistema

El sistema recibirá el nombre de “ONE JOB” el cual será visible desde la interfaz y el logotipo del producto. La Figura 16 muestra el logotipo de la aplicación.

Figura 16. Logotipo ONE JOB

Desde este momento cuando se hagan referencias sucesivas a "ONE JOB" se estará hablando del sistema implementado.

3.2.2 Funcionalidad del sistema

A continuación se expondrán los elementos que formarán parte del sistema a desarrollar para qué posteriormente se puedan realizar una especificación de requisitos acorde a cada elemento.

ONE JOB es una aplicación distribuida basada en el modelo cliente-servidor. Por un lado, la aplicación del cliente permite llevar a cabo tanto tareas de generación como de obtención de trabajos. La aplicación cliente deberá ejecutarse desde un dispositivo móvil basado en Android, y deberá tener disponible una conexión a Internet para poder establecer la comunicación con otros dispositivos.

El servidor por su parte es el segundo elemento del que se compone el sistema. Al igual que en el caso de la aplicación cliente, es imprescindible la conexión a Internet para que las comunicaciones puedan realizarse. Además, el servidor deberá tener acceso a una base de datos que almacenará toda la información referida a los clientes y que permite posteriormente realizar operaciones sobre ellos.

3.2.3 Beneficios del sistema

A la hora de desarrollar el sistema, se buscarán una serie de características que aporten valores positivos a los usuarios que hagan uso de ellas. Estas características que contiene el sistema son las siguientes:

- **Fácil Distribución**. Gracias al mercado de aplicaciones presente, el sistema es fácilmente accesible para el usuario pudiendo instalarse en el dispositivo cliente.
- **Instalación sencilla para el usuario**. El proceso de instalación del sistema en el dispositivo deberá ser simple para el usuario, y que no requerirá su intervención.
- **Movilidad**. El sistema se encuentra instalado en un dispositivo móvil, por lo que adquiere todos los beneficios presentes en este tipo de herramientas como el poder trasladarse de un lugar a otro y que el sistema siga siendo totalmente funcional.
- **Información independiente del dispositivo**. La información referente al usuario y que tiene que mostrar el sistema (currículo del usuario, trabajos generados) no se encuentra almacenada en el propio dispositivo, por lo que el usuario puede acceder a la información aunque su dispositivo no sea el que ha utilizado anteriormente.
- **Costes de producción reducidos**. Debido al bajo número de herramientas tecnológicas físicas que utiliza su coste no es elevado y llevarlo tiene su mayor coste en el desarrollo software del mismo.

3.2.4 Características de los usuarios

El sistema está desarrollado para todos los usuarios en general, ya que todos pueden asumir los roles de ofertar y recibir trabajos. Esta circunstancia provoca que el sistema se desarrolle de tal forma que no sea necesario un nivel técnico para poder utilizar todas las funcionalidades.

La única capacidad que el usuario debe tener es en el uso de la interfaz para poder utilizar correctamente todos los elementos de ésta. Para que esta circunstancia se realice será necesario desarrollar una interfaz sencilla e intuitiva y que cumpla con los patrones de diseño establecidos que favorezcan el uso por parte del usuario.

3.2.5 Particularidades y restricciones

Las particularidades y las restricciones del sistema son unas características del mismo que hay que tener en cuenta para el posterior proceso de desarrollo. Estas particularidades y restricciones son las siguientes.

- **Sin límite de usuarios.** No se impone un límite específico de usuarios que puedan registrarse en el sistema ni hacer uso del mismo a la vez.
- **Eficiencia de la aplicación cliente.** La eficiencia de la aplicación del cliente (tiempos de computo de las operaciones) viene dada en un gran porcentaje por el dispositivo en el que se está ejecutando. El dispositivo afecta a la aplicación cliente en términos de rendimiento (nivel de procesamiento del dispositivo) y autonomía (capacidad de la batería).
- **La eficiencia de la aplicación servidor.** La eficiencia del servidor a la hora de realizar las operaciones está afectada directamente por el número de usuarios registrados en la base de datos del sistema. Un número mayor de usuarios provoca un número mayor de operaciones.
- **Conexión a Internet de la aplicación cliente y del servidor.** La calidad y la velocidad de la conexión del servidor y la aplicación cliente afecta la eficiencia del paso de mensajes entre los dos elementos. Es crítica esta situación en el dispositivo móvil ya que en muchas ocasiones la conexión se realizará a través de una conexión inalámbrica móvil (siendo este tipo de conexión menos estable).

3.3 Especificación de requisitos

En esta sección se expondrán los requisitos que se deberán tener en cuenta y formarán una parte principal del posterior proceso de diseño. El formato de las tablas en las que se realizará la especificación de requisitos tendrá la siguiente estructura:

ID:	**Nombre:**	
Descripción:		
Prioridad: ☒ Esencial	☐ Deseado	☐ Opcional
Requisitos relacionados:		

Tabla 1 Especificación de requisitos

Los elementos que se pueden observar en la tabla son los siguientes:

- Un identificador de requisito para representarlo unívocamente.
- Un nombre del requisito que permita obtener una idea general sobre el contenido que tendrá el mismo.
- Una descripción del requisito, en el que se expondrá más específicamente cual será la función del mismo.
- El campo Prioridad representa el nivel de obligatoriedad de este requisito. Los valores que puede tomar son: Esencial (su realización es fundamental para el sistema), Deseado (sería beneficioso llevarlo a cabo) y Opcional (su no incorporación al sistema no afectaría al funcionamiento principal del mismo).

En cuanto a la nomenclatura del identificador de los requisitos se utilizará “F” o “NF” en el caso de que se trate de un requisito funcional o no funcional, respectivamente. Independientemente de esto, aparecerá una “C” en el caso de que el requisito sea de la aplicación cliente, “S” en el caso de que sea del servidor y “BD” para la base de datos. Por último se añade un número para diferenciar requisitos que pertenezcan a un mismo subtipo.

3.3.1 Requisitos funcionales

Los requisitos funcionales estarán divididos para la aplicación cliente, para el servidor y para la base de datos.

3.3.1.1 Aplicación cliente

ID: FC01	**Nombre:** Registro de un usuario
Descripción: El usuario será capaz de registrarse en el sistema. Para ello debe introducir su dirección de e-mail y una contraseña dos veces. La aplicación cliente será informada en el caso de que el proceso de registro se haya realizado correctamente o se haya producido un error.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 2 Requisito Registro de un usuario

ID: FC02	**Nombre:** Introducir datos login
Descripción: El usuario podrá introducir los datos que ha introducido en el registro para acceder al sistema identificado como ese usuario en particular. Los datos a introducir serán la dirección email y la contraseña.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC01	

Tabla 3 Requisito de Introducir datos login

ID: FC03	**Nombre:** Enviar datos registro
Descripción: La aplicación de usuario será capaz de enviar los datos que se han introducido en el registro al servidor.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC01	

Tabla 4 Requisito de enviar datos registro

ID: FC04	**Nombre:** Enviar datos login
Descripción: La aplicación de usuario será capaz de enviar los datos que se han introducido en el login al servidor para verificar la identidad del usuario.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC02	

Tabla 5 Requisito de enviar datos login

ID: FC05	**Nombre:** Recibir datos del registro
Descripción: La aplicación de usuario será capaz de recibir la verificación del servidor de que el registro se ha realizado correctamente.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC02	

Tabla 6 Requisito de recibir datos del registro

ID: FC06	**Nombre:** Recibir datos login
Descripción: La aplicación de usuario será capaz de recibir la verificación del servidor de que los datos introducidos en el proceso de login son correctos.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC02,FC04	

Tabla 7 Requisito de recibir datos login

ID: FC07	**Nombre:** Generar un trabajo
Descripción: El usuario será capaz de comenzar a generar un trabajo pulsando un botón que se encuentra en la interfaz.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC03	

Tabla 8 Requisito de generar un trabajo

ID: FC08	**Nombre:** Introducir datos al generar trabajo
Descripción: El usuario podrá introducir el campo que se relaciona con el trabajo, la descripción del empleo a generar, la dirección donde se realiza el trabajo y el título que tendrá.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC07	

Tabla 9 Requisito de introducir datos al generar trabajo

ID: FC09	**Nombre:** Envío de datos al generar trabajo
Descripción: La aplicación de usuario será capaz de enviar los datos sobre el trabajo generado al servidor una vez el propio usuario verifique los datos introducidos para generarlo.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC08	

Tabla 10 Requisito de envío de datos al generar trabajo

ID: FC10	**Nombre:** Seguimiento de los trabajos generados
Descripción: El usuario podrá ver los trabajos que ha generado hasta el momento	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC08,FC09	

Tabla 11 Requisito de seguimiento de los trabajos generados

ID: FC11	**Nombre:** Visualizar candidatos
Descripción: El usuario podrá ver los candidatos que ha seleccionado el sistema para el trabajo que ha generado.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 12 Requisito de visualizar candidatos

ID: FC12	**Nombre:** Visualizar información candidatos
Descripción: El usuario puede visualizar la información del contacto y la descripción del candidato al trabajo que ha generado.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC11	

Tabla 13 Requisito de visualizar información candidatos

ID: FC13	**Nombre:** Añadir datos currículum por primera vez.
Descripción: Se le dará la opción al usuario de introducir los datos de su currículum si todavía no los ha introducido. Entre la información que debe introducir encontramos: Una foto personal, su nombre y apellidos, nombre de empresa, su ubicación, el campo de su especialidad, la distancia que está dispuesta a viajar para trabajar, una descripción más específica del trabajo que se debe realizar y una forma de contacto con el usuario.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 14 Requisito de añadir datos currículum por primera vez

ID: FC14	**Nombre:** Modificar los datos del Curriculum
Descripción: El usuario podrá ser capaz de modificar los datos introducidos anteriormente en su curriculum.	
Prioridad: ☐ Esencial	☐ Deseado ☒ Opcional
Requisitos relacionados: FC13	

Tabla 15 Requisito de modificar los datos del curriculum

ID: FC15	**Nombre:** Aceptar ofertas de trabajo
Descripción: El usuario podrá ser capaz de aceptar los trabajos para los que ha sido seleccionado por el sistema, permitiendo que sus datos de contacto sean visibles para el generador del trabajo.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 16 Requisito de aceptar ofertas de trabajo

3.3.1.2 Servidor

ID: FS01	**Nombre:** El servidor deberá aceptar conexiones de los clientes
Descripción: El servidor deberá ser capaz de aceptar una conexión entrante por parte de la aplicación cliente. Además se deberá almacenar información acerca de los datos de conexión de la aplicación cliente (Dirección IP, puerto)	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 17 Requisito de aceptar conexiones de los clientes

ID: FS02	**Nombre:** El servidor debe almacenar los datos enviados por los usuarios
Descripción: El servidor deberá ser capaz de realizar una conexión a base de datos que le permita realizar las inserciones y consultas necesarias	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 18 Requisito de almacenar los datos enviados por los usuarios

ID: FS03	**Nombre:** Validar correo electrónico en el registro
Descripción: El servidor deberá validar que el correo enviado por la aplicación cliente en el registro no se encuentra ya en el sistema.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC02	

Tabla 19 Requisito de validar correo electrónico en el registro

ID: FS04	**Nombre:** Almacenar datos del registro
Descripción: El servidor deberá ser capaz de almacenar la información suministrada por la aplicación cliente en el proceso de registro	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC02	

Tabla 20 Requisito de almacenar datos del registro

ID: FS05	**Nombre:** Validar los datos del login
Descripción: El servidor deberá ser capaz de validar si la información suministrada en el proceso de login de un usuario es correcta.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FC03	

Tabla 21 Requisito de validar los datos del login

ID: FS06	**Nombre:** Almacenar los datos al generar el trabajo
Descripción: El servidor deberá ser capaz de almacenar la información suministrada por la aplicación cliente al generar un trabajo.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FS02	

Tabla 22 Requisito de almacenar los datos al generar el trabajo

ID: FS07	**Nombre:** Listar trabajos generados
Descripción: El servidor deberá ser capaz de enviar a la aplicación cliente la lista de todos los trabajos generados por el usuario;	
Prioridad: ☐ Esencial	☒ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 23 Requisito de listar trabajos generados

ID: FS08	**Nombre:** Realizar la selección de candidatos
Descripción: El servidor deberá ser capaz de seleccionar los candidatos correctos atendiendo a los datos suministrados en el proceso de inserción de curriculum y generación de trabajo. Estos datos serán el campo del trabajo y la posición geográfica donde se encuentra el mismo.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FS06	

Tabla 24 Requisito de realizar la selección de candidatos

ID: FS09	**Nombre:** Enviar ofertar a los usuarios
Descripción: El servidor deberá ser capaz de enviar las ofertas de trabajo a los candidatos potenciales seleccionados. La selección se realiza mediante el requisito FS08.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados: FS08	

Tabla 25 Requisito enviar ofertas a los usuarios

ID: FS10	**Nombre:** Existencia de curriculum
Descripción: El servidor deberá informar a la aplicación cliente de si el usuario que ha realizado el login tiene ya un curriculum asociado o no.	
Prioridad: ☐ Esencial	☒ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 26 Requisito de existencia de curriculum

ID: FS11	**Nombre:** Concurrencia del servidor
Descripción: El servidor deberá ser capaz de aceptar múltiples peticiones por parte de aplicaciones clientes de forma concurrente.	
Prioridad: ☐ Esencial	☒ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 27 Requisito de concurrencia del servidor

3.3.1.3 Base de datos

ID: FBD01	**Nombre:** Conexión con el servidor
Descripción: La base de datos debe permitir que el servidor realice una conexión ella así atender las peticiones de éste.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 28 Requisito de conexión con el servidor

ID: FBD02	**Nombre:** Existencia de un control de accesos
Descripción: Para mejorar la seguridad la base de datos deberá pedir credenciales al realizar una conexión.	
Prioridad: ☐ Esencial	☒ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 29 Requisito de existencia de un control de accesos

ID: FB03	**Nombre:** Ejecución de consultas
Descripción: La base de datos deberá ser capaz de ejecutar sentencias de consulta de la información, de inserción y de actualización de las sentencias que el servidor requiera ejecutar.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 30 Requisito de ejecución de consultas

ID: FB04	**Nombre:** Conexiones activas BBDD
Descripción: La base de datos deberá ser capaz de mantener activas varias conexiones con el servidor.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 31 Requisito de conexiones activas de la base de datos

3.3.2 Requisitos no funcionales

ID: NF01	**Nombre:** Rendimiento
Descripción: El funcionamiento del sistema deberá ser independiente del número de usuarios conectados al mismo.	
Prioridad: ☐ Esencial	☒ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 32 Requisito de rendimiento

ID: NF02	**Nombre:** Disponibilidad	
Descripción: El sistema deberá mantenerse activo y disponible para que el usuario pueda hacer uso de él.		
Prioridad: ☐ Esencial	☒ Deseado	☐ Opcional
Requisitos relacionados:		

Tabla 33 Requisito de disponibilidad

ID: NF03	**Nombre:** Sistema operativo del cliente	
Descripción: El sistema operativo del cliente será Android y estará optimizado para un dispositivo de aproximadamente 5 pulgadas.		
Prioridad: ☐ Esencial	☒ Deseado	☐ Opcional
Requisitos relacionados:		

Tabla 34 Requisito de sistema operativo del cliente

ID: NF04	**Nombre:** Lenguaje de programación del servidor	
Descripción: El lenguaje de programación del servidor será C.		
Prioridad: ☐ Esencial	☒ Deseado	☐ Opcional
Requisitos relacionados:		

Tabla 35 Requisito de lenguaje de programación del servidor

ID: NF05	**Nombre:** Base de datos seleccionada	
Descripción: La base de datos utilizada será MySQL		
Prioridad: ☐ Esencial	☒ Deseado	☐ Opcional
Requisitos relacionados:		

Tabla 36 Requisito de base de datos seleccionada

ID: NF06	**Nombre:** Concurrencia
Descripción: La implementación de la concurrencia en el servidor se realizará por medio de procesos ligeros o hilos.	
Prioridad: ☐ Esencial	☒ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 37 Requisito de concurrencia en el servidor

ID: NF07	**Nombre:** Comunicación
Descripción: La comunicación entre la aplicación cliente y el servidor se realizará por medio de sockets TCP.	
Prioridad: ☒ Esencial	☐ Deseado ☐ Opcional
Requisitos relacionados:	

Tabla 38 Requisito de concurrencia en el servidor

4. Diseño

En este capítulo se mostrará el diseño realizado en base al análisis del capítulo anterior. Además de esto, se pretende sentar las bases de la futura implementación.

4.1 Arquitectura básica del sistema

El sistema a desarrollar utilizará la arquitectura denominada "cliente servidor". En este modelo de sistema distribuido encontramos elementos cuyas tareas son claramente diferenciables, encontramos proveedores de los recursos denominados servidores y los demandantes de esos mismos recursos que se llaman clientes. Tanto los servidores como los clientes tienen claramente diferenciadas sus tareas asociadas, que van acorde con sus capacidades. Normalmente el que inicia la comunicación y mantiene el papel activo es el denominado cliente que realiza peticiones al elemento más pasivo que se establece como el servidor y que es el encargado de gestionar esas peticiones.

En el sistema a desarrollar, nos encontramos con un único servidor y con tantas aplicaciones clientes como aplicaciones del sistema estén instaladas en los dispositivos. El almacenamiento de la información se realiza por medio de una base de datos que tiene conectada el servidor y que le permite procesar y gestionar las peticiones realizadas por el cliente. La Figura 17 muestra la arquitectura cliente-servidor del sistema.

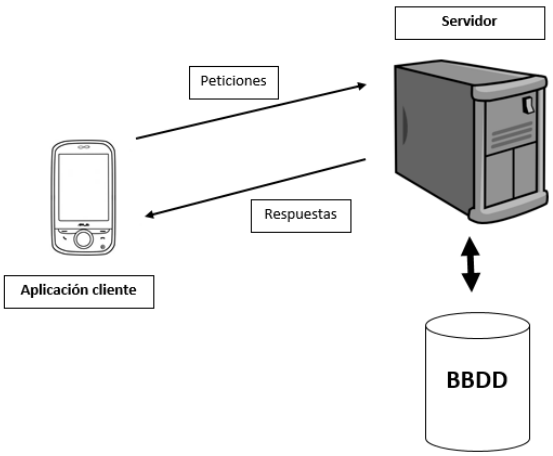

Figura 17. Arquitectura cliente-servidor del sistema

4.2 Flujo de utilización del sistema

En esta sección se mostrará de qué forma se produce la utilización del sistema por parte del usuario y cuáles son los pasos que se producen desde que un usuario genera un trabajo hasta que se pone en contacto con un candidato potencial y paralelamente, como un usuario inserta un curriculum y pasa a ser un candidato potencial de un trabajo.

La Figura 18 muestra el flujo de utilización del sistema por parte de los usuarios. Como puede observarse el usuario 1 genera un trabajo que mediante el proceso de selección se establece al usuario 2 como candidato potencial al mismo. Una vez el Usuario 2 recibe la oferta del trabajo generado la acepta, permitiendo que la información de su contacto sea enviada al usuario 1 (generador del trabajo).

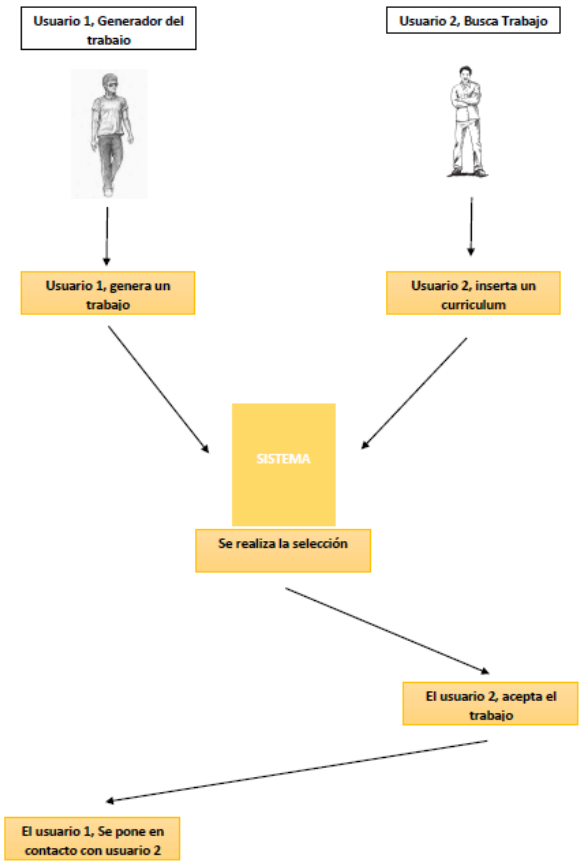

Figura 18. Flujo de utilización del sistema

4.3 Justificación del software

En esta sección se expondrán las decisiones tomadas en cuanto a la utilización del software para llevar a cabo el sistema, realizando una justificación de la misma.

4.3.1 Sistema operativo de la aplicación cliente

Entre los distintos tipos de sistemas operativos disponibles para desarrollar la aplicación cliente se decide realizarlo con el sistema operativo Android. Esta decisión se toma debido a ciertas características positivas que aporta el desarrollo con en este sistema.

En primer lugar la implantación que tiene este sistema operativo en los smartPhones de todo el mundo permite que el número potencial de usuarios que tenga el sistema sea muy elevado. Que el número de usuarios sea tan elevado permite también mejorar el producto, ya que a mayor número de usuarios el tráfico de trabajos generados y de gente que pueda suplir ese trabajo también será mayor. Otra característica positiva es que nos permite realizar uso de todas las capacidades que tiene el dispositivo como la comunicación en Internet o el servicio de localización que nos permitirá llevar a cabo las funcionalidades del sistema.

Además de esto, Android tiene una serie de APIs que nos permiten compatibilizar las comunicaciones independientemente del sistema operativo del servidor, punto muy importante porque el sistema operativo del servidor no será el mismo que el de la aplicación cliente como ya se verá posteriormente.

Desde el punto de vista del desarrollador, Android concede un gran número de facilidades desde un entorno estructurado donde desarrollar las aplicaciones hasta una cuantiosa documentación.

4.3.2 Lenguaje de programación del servidor

El servidor del sistema encargado de tratar las peticiones de las aplicaciones clientes estará programado en lenguaje C. Esta decisión se toma debido a distintas características de este lenguaje que aportan beneficios al sistema.

En primer lugar, este lenguaje de programación tiene implementadas bibliotecas que permiten la comunicación a través de Internet y que permitirá por tanto recibir mensajes de las aplicaciones clientes y enviar respuestas. Esta circunstancia es un punto clave a la hora de que el sistema este implementado siguiendo el modelo de cliente-servidor. Otra característica fundamental es la facilidad para realizar una conexión con la base de datos y el posterior tratamiento de la información obtenida. Debido al gran número de peticiones que el servidor realizará a la base de datos es imprescindible que esta acción se ejecute de forma eficiente y que no provoque pérdidas de rendimiento.

Atendiendo a las características propias del lenguaje C, encontramos que permite trabajar en muchas ocasiones como un lenguaje cercano al de bajo nivel, lo cual favorece el desarrollo de programas que interactúan con el nivel de hardware para gestionar por ejemplo memoria o procesos.

4.3.3 Base de datos

La base de datos estará gestionada utilizando MySQL. Este sistema de gestión de bases de datos permitirá realizar las funcionalidades que el sistema necesita de una forma correcta gracias a sus distintas características.

En primer lugar su inserción en la funcionalidad de un programa se realiza de forma sencilla y eficiente por lo que es favorable para que el servidor lo utilice en el almacenamiento y gestión de la información.

Este sistema permite también mantener varias conexiones activas lo que como se verá posteriormente permite que el servidor realice ciertas funcionalidades que serían imposibles si la capacidad de conexión múltiple de MySQL.

4.3.4 Comunicaciones (Sockets)

Las comunicaciones son un punto clave del sistema y debido a que nos encontramos distintos sistemas operativos para el cliente que para el servidor también es un punto delicado.

Esta circunstancia se resuelve correctamente gracias a los sockets, los cuales están implementados para ambos sistemas operativos permitiendo la compatibilidad de la recepción y envío de información entre los elementos que componen el sistema. Además de esto, los sockets permiten el envío y recepción de numerosos tipos de datos. Esta circunstancia será muy favorable gracias a que durante la ejecución del sistema se tendrán que trabajar con distintas informaciones que serán de distintos tipos.

En cuanto al protocolo de transporte seleccionado para el posterior proceso de implementación se selecciona TCP (Transmission Control Protocol). Este protocolo orientado a conexión permite tener la seguridad de que la información que se envíe a través de los sockets llegará al destinatario, sin errores y en el orden en el que el programa lo haya enviado. Esta decisión se toma debido a la importancia que tiene en el sistema que alguno de los mensajes se pierda o sea recibido con errores. Además de esto, que los mensajes lleguen al destinatario en el mismo orden en el que fueron enviados por el emisor es un punto clave para un sistema que implementa protocolos de comunicación.

4.4 Diseño de la aplicación cliente

En esta sección se mostrará la solución al diseño de la aplicación cliente, observando las funcionalidades que deberán ser implementadas posteriormente y el flujo de ejecución y diseño de la interfaz de la aplicación.

4.4.1 Funciones

La aplicación cliente tendrá que ser capaz de realizar las siguientes funcionalidades en el sistema:

- Ser el elemento que interactúe directamente con el usuario y que permita invocar las funcionalidades que éste precise.
- Realizar la petición de generar trabajo con los parámetros seleccionados por el usuario.
- Listar los trabajos que ya ha generado ese usuario, pudiendo observar los candidatos que el sistema ha seleccionado para los mismos.
- Insertar un curriculum en el sistema acorde con parámetros personalizados del usuario.
- Listar los trabajos disponibles acordes con los datos del curriculum.
- Aceptar un trabajo si éste aparece listado como un trabajo para el que se cumplen los requisitos.

Todas estas funcionalidades deberán tener un diseño de interfaz de usuario acorde con la funcionalidad requerida y que permita al usuario interactuar con la aplicación cliente de manera sencilla. Este diseño de interfaz se mostrará en secciones posteriores.

4.4.2 Diagrama de actividades

Para realizar el diagrama que represente a todas las actividades de la aplicación cliente vamos a estudiar en primer lugar de qué actividades se compone el sistema para poder posteriormente realizar la asociación entre ellas. La aplicación cliente se compone de las siguientes actividades:

- **Actividad Inicialización:** Esta actividad es la encargada de realizar la conexión con el servidor mediante sockets. Una vez realizada la conexión se utilizará ésta misma para realizar todas las comunicaciones de envío y recepción de datos que se precisen con el servidor.

- **Actividad Login:** Esta actividad permitirá al usuario introducir la información para identificarse en el sistema. Para ello se debe habilitar la opción de introducir su email y la contraseña. Una vez el usuario tenga introducida esta información, la actividad se encargará de enviar esa información al servidor, que deberá posteriormente validarla.
- **Actividad Registro:** La funcionalidad de registrarse en el sistema será el cometido de esta actividad. Deberá habilitarse en la interfaz los elementos que permitan introducir la información por parte del usuario así como el posterior envío al servidor de la información.
- **Actividad Principal:** Esta actividad es la que maneja mayor información de la aplicación del sistema. En primer lugar se podrá ver si se tiene ya insertado un curriculum en el sistema y en el caso de que esto se haya realizado ya, se mostrarán todos los trabajos para los que el sistema ha seleccionado al usuario. También se puede empezar a generar trabajos en esta actividad y a listar todos los que el usuario haya generado antes. El flujo de ejecución de esta actividad es el siguiente:
    1. Primero realiza la petición en el servidor para conocer la existencia o no de curriculum para ese usuario.
    2. Posteriormente lista los trabajos generados por el usuario anteriormente en el caso de que los hubiera.
    3. Lista los trabajos en los que el usuario ha sido seleccionado como candidato por el sistema en el caso de que se haya realizado un curriculum.
- **Actividad Curriculum:** Esta actividad genera la primera pantalla que se muestra cuando el usuario decide insertar un curriculum. En este primer paso a la hora de rellenar el curriculum, se introduce información como el nombre, los apellidos y el nombre de la empresa en el caso de que el usuario tuviera asociada una que es la que puede llevar a cabo el trabajo.
- **Actividad Curriculum2:** El segundo paso para introducir el curriculum en el sistema se muestra a través de esta actividad. En esta ocasión se deberá insertar información de la ubicación donde reside el usuario. Esta

información se dará en forma de longitud y latitud y deberá ser enviada posteriormente al servidor.

- **Actividad Curriculum 3:** Esta actividad forma parte de la tercera pantalla a la hora de insertar un curriculum. En este caso la información a suministrar será la formación del usuario seleccionando por tanto un campo entre los presentes ya en el sistema. También se podrá añadir una descripción personal que sirva para definir mejor las capacidades del usuario. Por último se deberá señalar la distancia en kilómetros que el usuario está dispuesto para trabajar. Este punto es muy importante para realizar la posterior selección del candidato al trabajo.
- **Actividad Curriculum 4:** Esta es la última actividad de la que se compone la inserción de un curriculum por parte de un usuario. En este caso se introducirá la información referente a la futura forma de contacto con el usuario, pudiendo seleccionar un teléfono, un email y otra forma de contacto. Esta actividad también es la encargada de realizar el envío de toda la información que el usuario ha ido suministrando a través de las actividades de curriculum. Por tanto se enviará al servidor:
    - Nombre del usuario.
    - Apellidos del usuario.
    - Nombre de empresa (en el caso de que lo hubiera).
    - Latitud y longitud de la ubicación donde reside el usuario.
    - Campo especializado del usuario.
    - Distancia que el usuario está dispuesto a viajar para trabajar.
    - Descripción personal del usuario.
    - Teléfono de contacto con el usuario.
    - Email del usuario.
    - Otra forma de contacto con el usuario.
- **Actividad GenerarTrabajo:** Esta actividad es la encargada de permitir al usuario introducir la información referente para generar un trabajo. Esta información es un título para nombrar al trabajo, una dirección del lugar donde se debería llevar a cabo el trabajo, el campo especializado que se pide para un usuario potencial que pueda realizar el trabajo y una descripción

del trabajo que se debería realizar donde también se puede incluir temas como horarios y salario disponible. Esta actividad también es la encargada de realizar el envío de la información al servidor. La información que es enviada es aquella que se ha mencionado anteriormente (título, dirección, campo, descripción) además de la ubicación del usuario cuando genera el trabajo.

- **Actividad VisualizarTrabajo:** Esta actividad permite visualizar los candidatos que el sistema ha seleccionado para el trabajo que se ha generado. Estos candidatos anteriormente también han tenido que realizar la aceptación del trabajo.
- **Actividad VisualizarCandidato:** De la actividad anterior de visualizar trabajo obteníamos una lista de candidatos potenciales para el trabajo generado. Mediante la actividad de VisualizarCandidato podemos obtener la información asociada al candidato en particular como su nombre completo, la descripción personal que introdujo a la hora de realizar el curriculum, y la forma de ponerse en contacto con él. De esta forma el usuario que ha generado el trabajo ya puede utilizar este contacto para que este candidato le realice el trabajo.
- **Actividad VisualizarOferta:** Esta actividad permite visualizar las ofertas que el sistema selecciona para un usuario en particular que ya ha rellenado un curriculum. El usuario podrá ver la dirección del trabajo y la descripción del trabajo a realizar. En esta actividad también se habilita mediante la interfaz la posibilidad de aceptar ese trabajo, lo que provoca que la información personal del usuario sea enviada al generador del trabajo como candidato potencial del mismo.

Estas actividades forman parte del flujo de ejecución de la aplicación cliente como se muestra en la Figura 19. En un principio observamos la actividad del login desde la que podremos o bien realizar un registro (actividad Registro) o introducir los datos para identificarse e ir a la siguiente actividad, la actividad Principal. La actividad Principal permite gestionar las funciones de la aplicación para que el usuario pueda interactuar con ellas:Generar un trabajo, Insertar un curriculum, Listar los trabajos generados y Listar las ofertas asociadas.

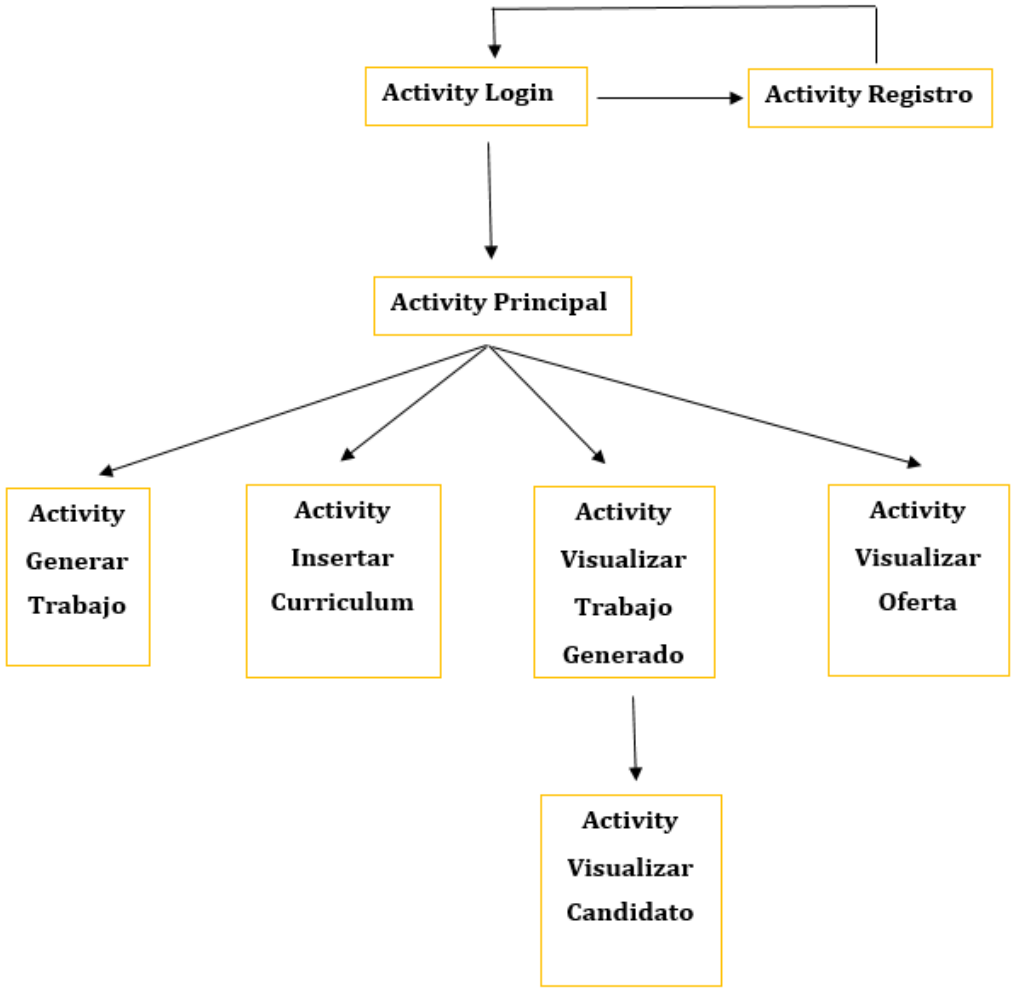

Figura 19. Flujo de ejecución de las actividades

4.4.3 Diseño de la interfaz

La interfaz de la aplicación cliente tiene un papel fundamentas para que el usuario pueda interactuar correctamente con el sistema y haga uso de todas sus funcionalidades. Para que esto se pueda realizar, la interfaz seguirá los patrones de diseño básicos acordes con la funcionalidad que desean ofrecer. Las actividades más importantes tienen el siguiente diseño de interfaz:

- **Actividad Login:** El diseño de interfaz de esta actividad se muestra en la Figura 20. Como puede observarse, se habilita la posibilidad de introducir los datos que validen al usuario tales como el nombre de usuario y la

contraseña. También se permite acceder a la opción de registrarse, para que el usuario pase a formar parte del sistema y pueda realizar la identificación correctamente.

Figura 20. Interfaz del Login

- **Actividad Registro:** El diseño de interfaz de esta actividad se muestra en la Figura 21.

Figura 21. Interfaz del registro

La interfaz permite introducir toda la información necesaria para que el proceso de registro se realice de forma correcta (nombre de usuario y contraseña).

- **Actividad Principal:** Esta actividad es la que más funcionalidades y opciones aporta al usuario y por lo tanto su interfaz contiene más elementos que el resto de actividades. La actividad se divide en dos pestañas, una para buscar trabajo y otra para generar los trabajos. El diseño de la interfaz se puede observar en la Figura 22 y en la Figura 23.

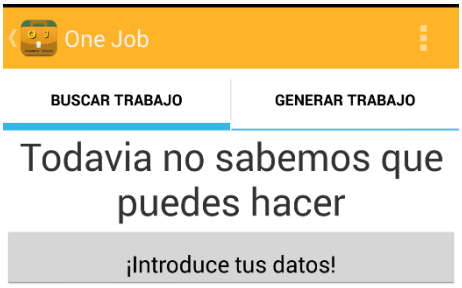

Figura 22. Interfaz de la actividad principal 1

Figura 23. Interfaz de la actividad principal 2

- **Actividades del curriculum:** Las actividades que permiten la introducción de un curriculum en el sistema tienen el diseño que se puede observar en las figuras 24, 25, 26 y 27.

Figura 24 Interfaz Curriculum 1

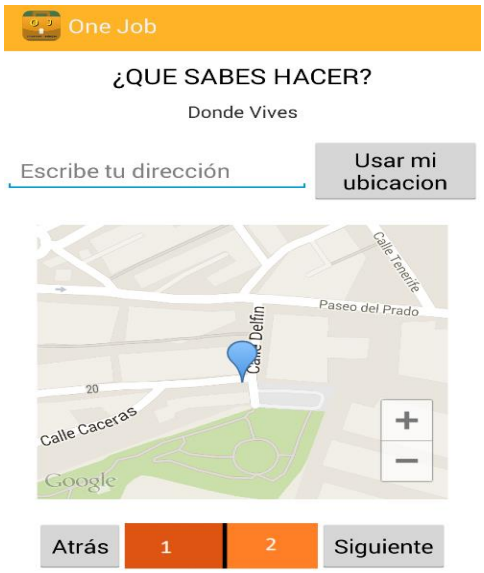

Figura 25 Interfaz Curriculum 2

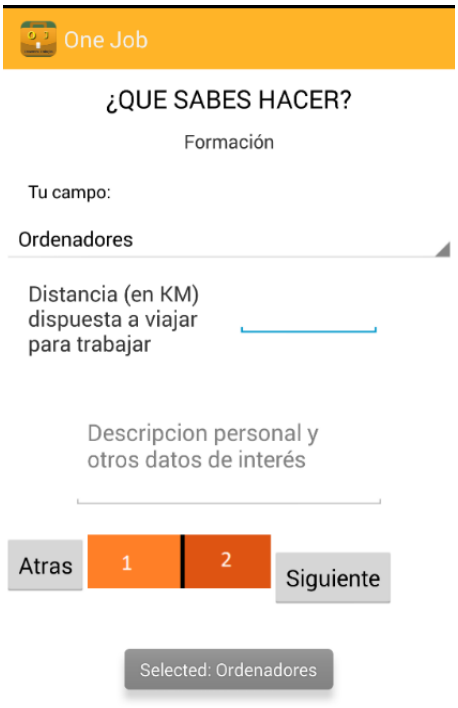

Figura 26 Interfaz curriculum 3

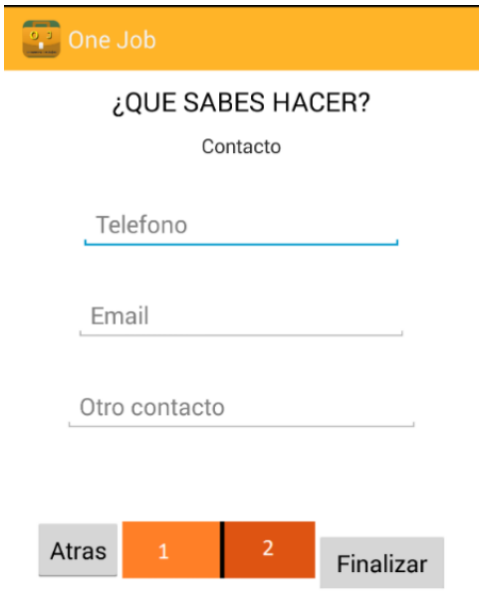

Figura 27 interfaz curriculum 4

- **Actividades GenerarTrabajo:** Esta actividad tiene el siguiente diseño de interfaz:

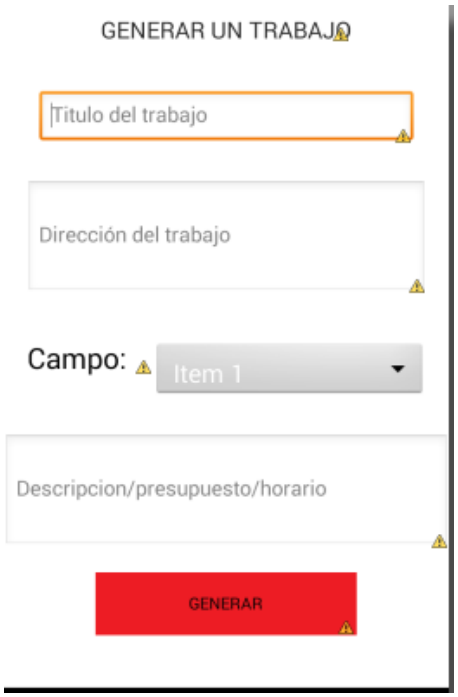

Figura 28. Interfaz generar trabajo

4.5 Diseño del servidor

El servidor será el encargado de tratar las peticiones que le lleguen de las aplicaciones cliente. Deberá por tanto poseer todas las funcionalidades que puedan ser requeridas por la aplicación incluyendo para ello la interacción con la base de datos.

4.5.1 Funcionalidades

Entre las funcionalidades que el servidor debe ser capaz de realizar dentro del sistema destacamos las siguientes:

- Aceptar conexiones de las aplicaciones clientes.
- Intercambiar mensajes con la aplicación cliente pudiendo contener estos mensajes distintos tipos de datos.
- Autenticar los datos que el usuario introduce en el proceso de Login.

- Realizar registros de usuarios en el sistema.
- Permitir la inserción de un curriculum.
- Visualizar si el usuario ha insertado un curriculum anteriormente.
- Permitir insertar un trabajo nuevo en el sistema.
- Realizar la selección de los candidatos potenciales a un trabajo generado.
- Realizar un listado de todos los trabajos generados por un usuario.
- Realizar un listado de todos los trabajos para los que un usuario es candidato potencial.
- Realizar el cálculo de distancias dadas las longitudes y latitudes de dos puntos geográficos.
- Realizar conexiones concurrentes a la base de datos.

4.5.1 Esquema del diseño

El servidor sigue un esquema de ejecución para tratar y responder las peticiones de las aplicaciones cliente. Este esquema es diseñado conjuntamente con la aplicación cliente ya que es necesario que ambos sean conocedores de esta estructura para que las peticiones y las respuestas sean las correctas. El servidor seguirá flujo de ejecución de la Figura 29.

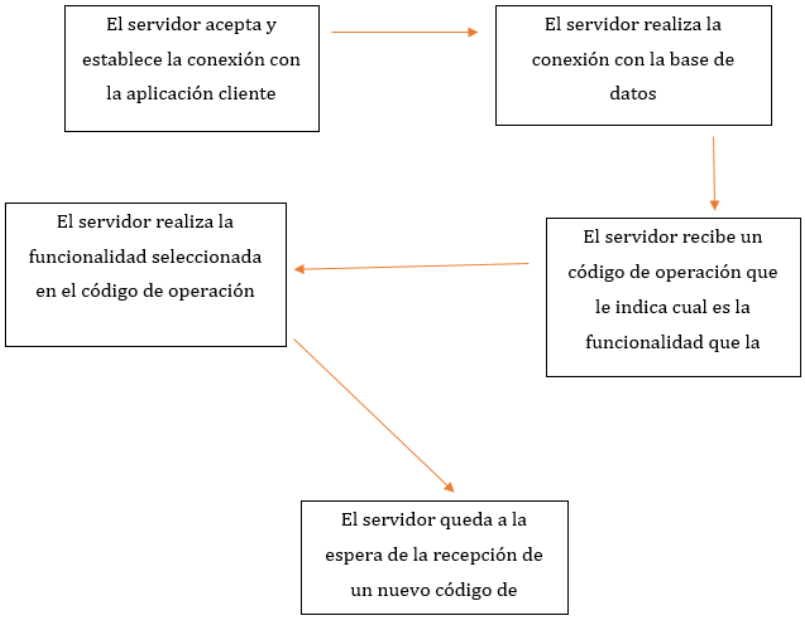

Figura 29. Flujo de ejecución de funcionalidades en el servidor

El flujo de ejecución que se muestra en la Figura 29 describe los pasos que realiza el servidor:

- Paso 1: El servidor recibe peticiones de conexión de la aplicación cliente aceptándolas.
- Paso 2: Una vez se acepta la conexión de la aplicación cliente, el servidor realiza la conexión con la base de datos.
- Paso 3: El servidor recibe un código de operación. Mediante los códigos de operación, el servidor reconoce la funcionalidad que la aplicación cliente está requiriendo.
- Paso 4: Una vez la funcionalidad que la aplicación cliente ha seleccionado se ha realizado, el servidor queda a la espera de que la aplicación cliente vuelva a necesitar otra funcionalidad (envíe un nuevo código de operación).

4.5.2 Diseño de la base de datos

La base de datos permitirá almacenar y tratar toda la información de los usuarios que contiene el sistema. El diseño de la base de datos debe hacerse de manera que permita llevar a cabo todas las funcionalidades que el servidor y el cliente requieren, haciéndolo además de una forma eficiente. La estructura de la base de datos se muestra en la Figura 30.

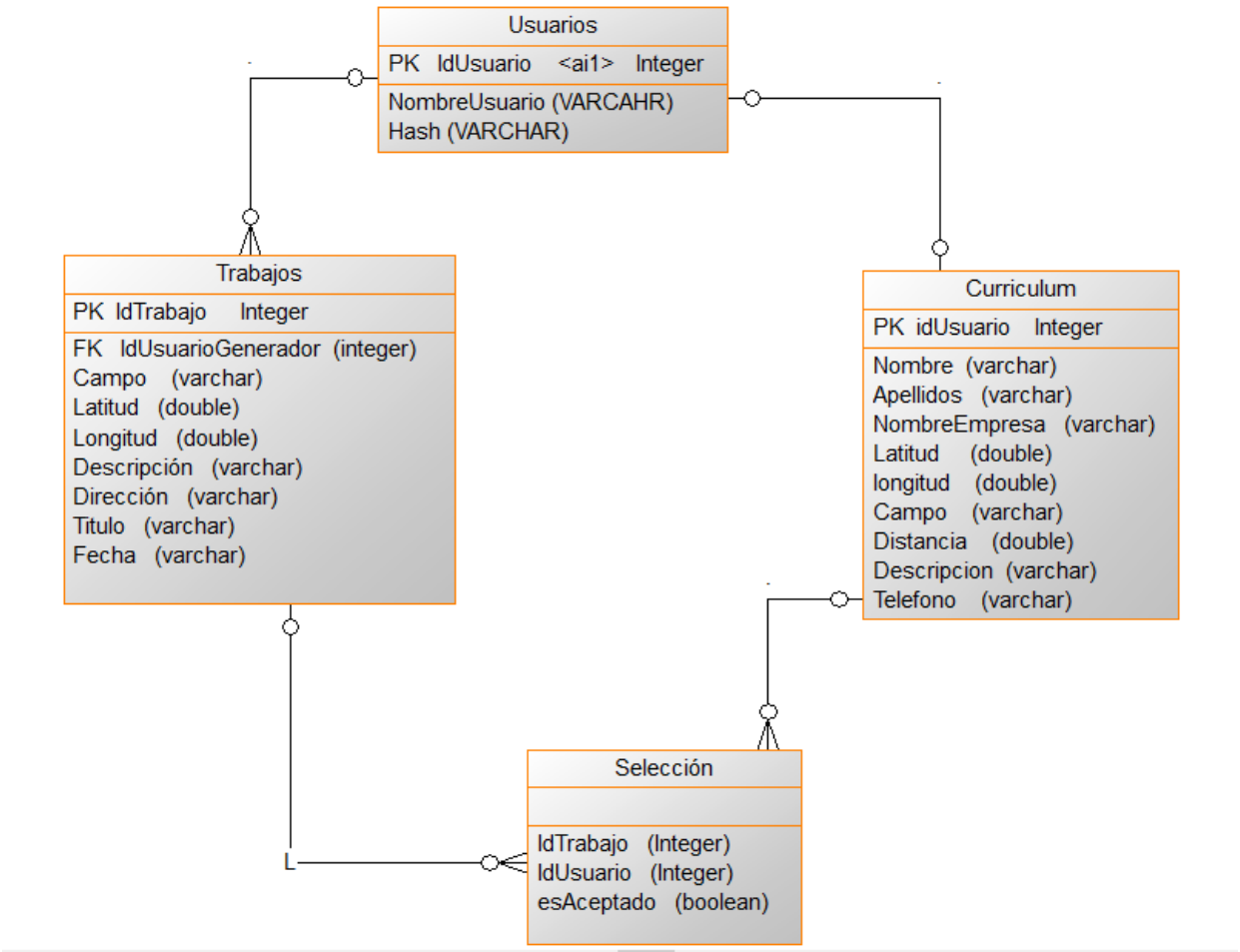

Figura 30. Diagrama de la Base de datos

La base de datos se compone de cuatro tablas: Trabajos, Usuarios, Curriculum y Seleccion. En la tabla Usuarios se almacenará la información referente a la validación de los usuarios en el sistema. La clave primaria de esta tabla es un identificador de usuario (id) que se genera automáticamente y que permitirá realizar el resto de consultas identificando unívocamente al usuario que las realiza.

La tabla Trabajos almacena toda la información de los trabajos que se han generado, identificado cada usuario mediante el identificador de usuario de la tabla usuarios e identificando cada fila de la tabla mediante un id de trabajo que también se genera automáticamente.

Los datos referentes a los curriculums insertados por los usuarios se almacenan en la tabla Curriculum. El id de usuario identifica cual es el usuario que ha insertado ese curriculum y toda la información relacionada con éste.

Por último la tabla de Seleccion almacena la información que genera el proceso de selección de candidato realizada por el servidor. Con esta tabla se pueden saber que candidatos potenciales están asociados a un trabajo y si estos candidatos han aceptado ya el trabajo o no.

4.6 Protocolo de servicio

En esta sección se diseñará la estructura y el formato de los mensajes que se enviarán la aplicación cliente y el servidor para las distintas funcionalidades del sistema.

A la hora de realizar el diseño de este apartado vamos a tener en cuenta buscar la mayor eficiencia en el envío de la información, evitando el mayor tráfico en la red posible y reduciendo las esperas de envíos de mensajes tanto para la aplicación cliente como para el servidor. Para conseguir esta eficiencia se decide concatenar la información con el mayor grado posible de forma que el número de mensajes que son necesarios enviar para la comunicación se vean claramente reducidos. En conclusión se realiza mayor envío de información por mensaje en lugar de enviar una cantidad elevada de mensajes con un contenido menor.

En cuanto a la concatenación de la información se ha decidido incluir un carácter especial entre los distintos elementos que se quieran enviar juntos. El elemento que se inserta entre los distintos valores y que permite la concatenación será '#'. A continuación se mostrará el protocolo de servicio de cada funcionalidad señalando como se produce la comunicación entre el cliente y el servidor.

4.6.1 Proceso de registro

En el proceso de registro, al igual que el resto de funcionalidades, el cliente envía en primer lugar el código de operación (en este caso el código de operación

del registro es 1). Es necesario que la aplicación realice el envío del nombre de usuario en primer lugar para comprobar que no existe ya ese nombre de usuario en el sistema. El protocolo de servicio más detallado puede verse en la Figura 31.

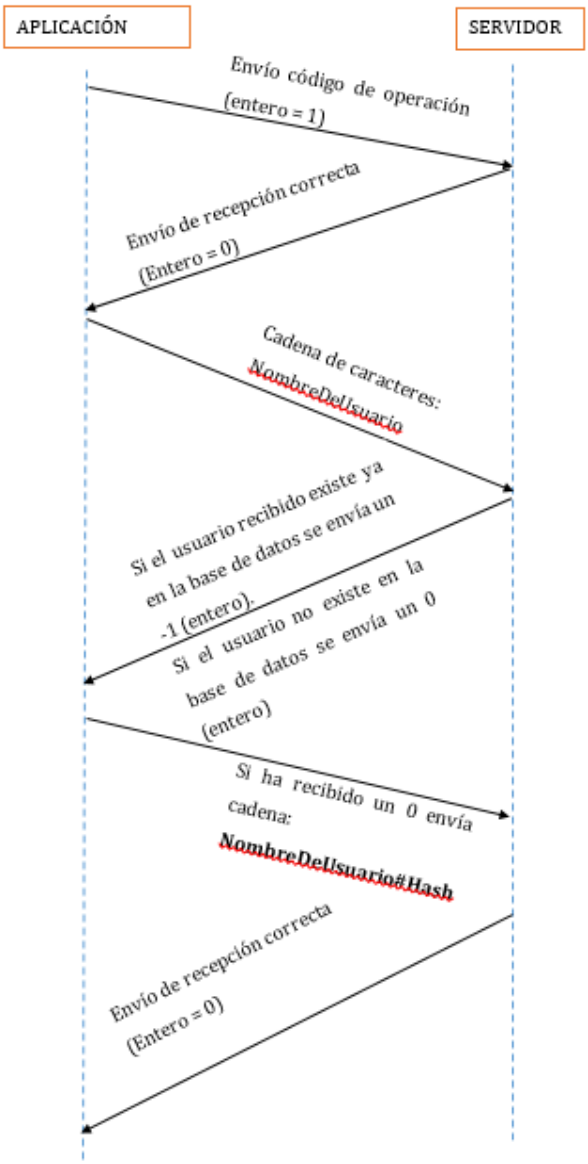

Figura 31. Protocolo de registro

4.6.2 Proceso de Login

El protocolo de servicio entre el cliente y el servidor cuando se realiza el proceso de Login se observa en la Figura 32.

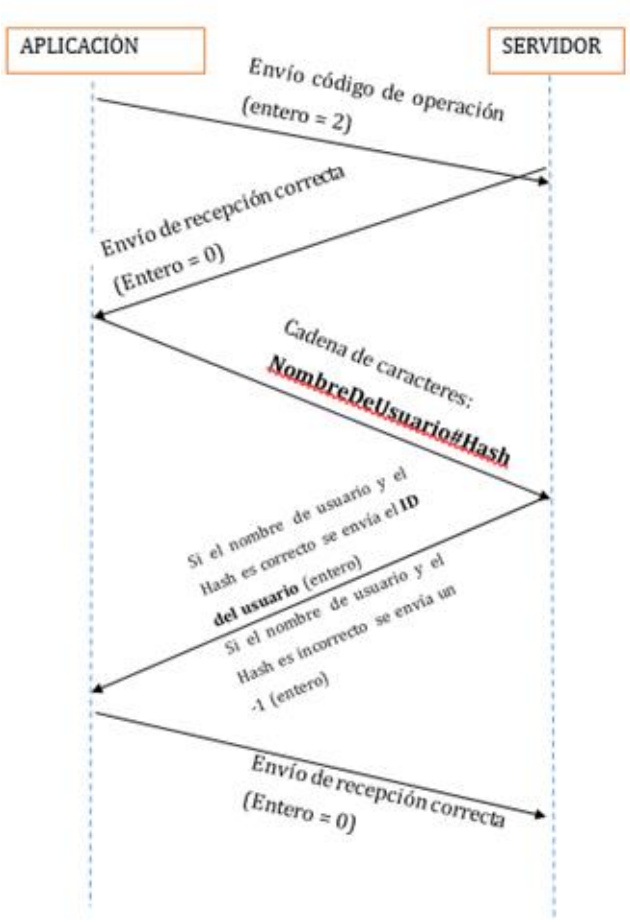

Figura 32. Protocolo de Login

4.6.3 Proceso de Insertar Curriculum

En el proceso de insertar un curriculum tiene asociado el código de operación 3. La cadena de caracteres que la aplicación cliente envía con la información del curriculum, se encuentra concatenada utilizando el carácter especial '#' como se ha mencionado anteriormente. El protocolo de servicio detallado se muestra en la Figura 33.

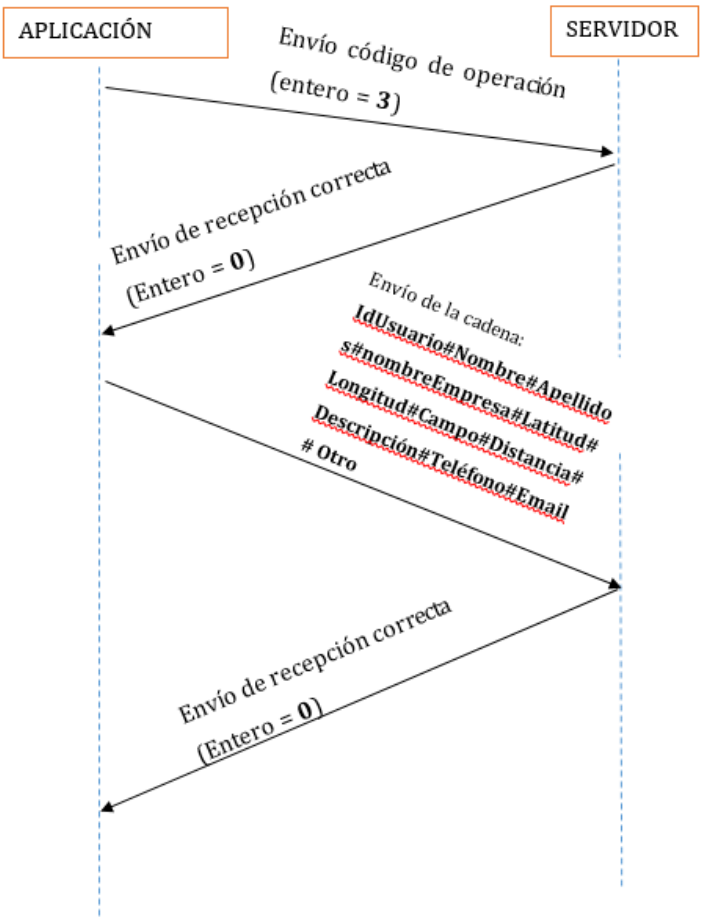

Figura 33 Proceso de insertar curriculum

4.6.4 Proceso de generar trabajo

En este caso el código de operación para realizar la generación de un nuevo trabajo es el número 4. Al igual que en el caso anterior, la información referente al nuevo trabajo generado se encuentra concatenado utilizando la técnica anteriormente mencionada. El protocolo de servicio cuando el usuario realiza esta funcionalidad se muestra en la Figura 34.

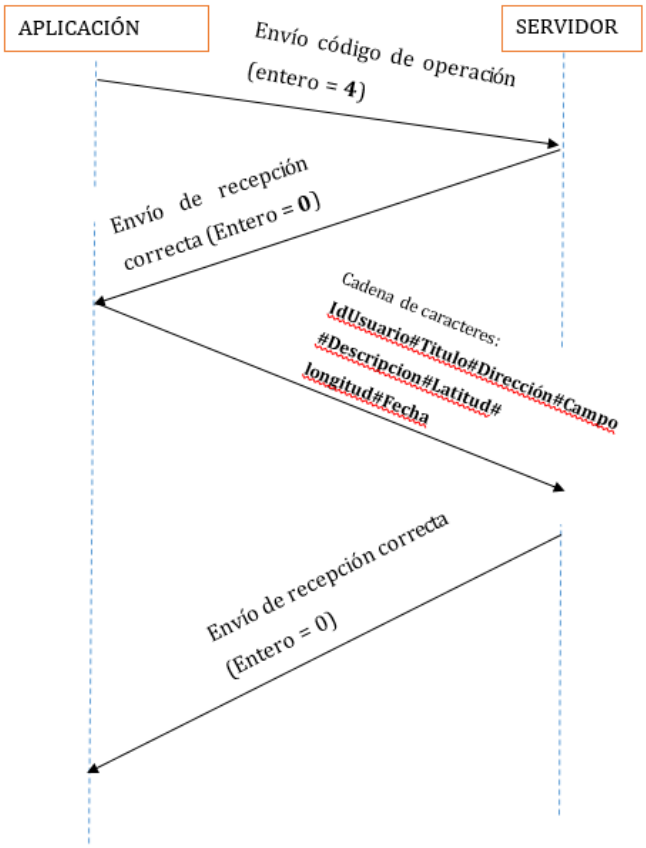

Figura 34. Protocolo de generar trabajo

4.6.5 Visualizar la existencia de curriculum

Para la funcionalidad de conocer la existencia de un curriculum para un usuario específico se utiliza el protocolo de servicio que muestra la Figura 35.

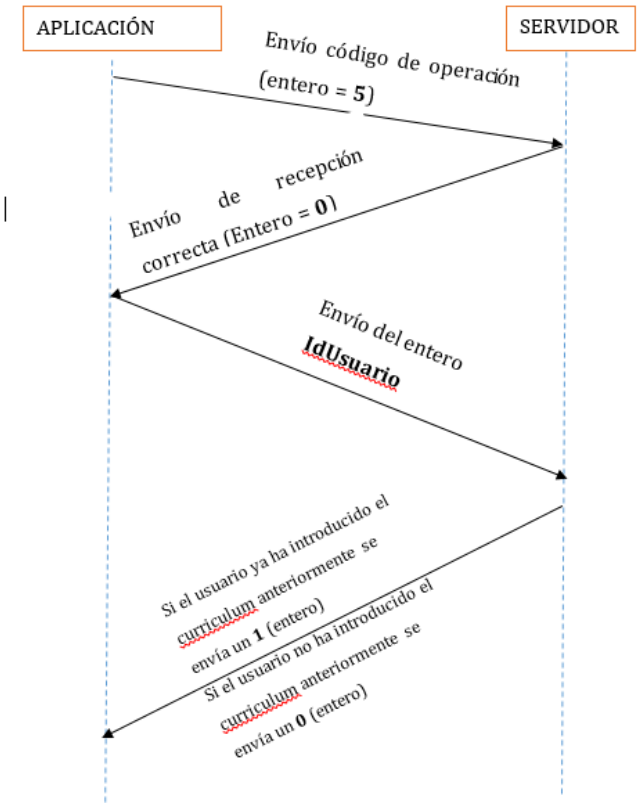

Figura 35. Protocolo de existencia de curriculum

4.6.6 Listar trabajos generados

Cuando se lista cualquier una serie de elementos en el sistema, siempre se procede de la misma forma. En primer lugar el servidor envía el número de elementos que tiene el sistema de la lista que se quiere listar (por ejemplo número de trabajos generados por un usuario específico) y a continuación se realizan tantos envíos con la información como elementos tenga la lista. Este es el caso por ejemplo de listar los trabajos generados cuyo protocolo se muestra en la Figura 36.

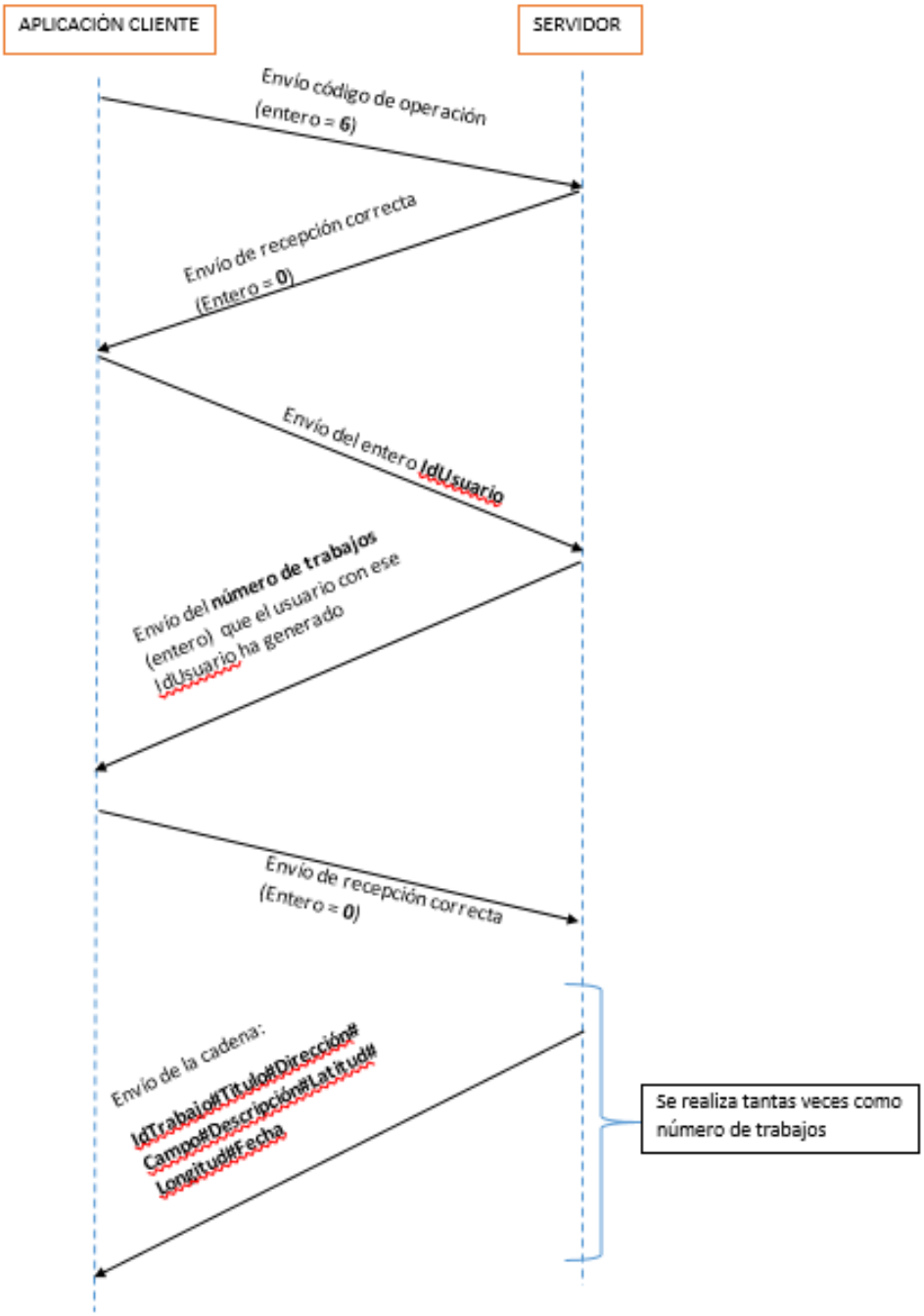

Figura 36. Protocolo de listar trabajos generados

5. Implementación

En este capítulo se mostrará la construcción del sistema mediante la codificación de los elementos que lo compone, que han sido diseñados en el capítulo anterior. Como se ha mencionado en capítulos anteriores, el sistema contiene una arquitectura de cliente-servidor por lo que se realizaran dos secciones, una para la implementación del cliente y otra para la implementación del servidor.

5.1 Implementación Aplicación cliente

En esta sección se estudiará la implementación de la aplicación cliente mostrando para ello el código de las partes más representativas y utilizadas de la aplicación y que tienen un papel fundamental para obtener la funcionalidad requerida.

5.1.1 Conexión con el servidor

La conexión con el servidor se realiza, como hemos estudiado en capítulos anteriores mediante sockets. Esta conexión se realiza en el momento en el que la aplicación se abre en la actividad nombrada Inicialización. En esta conexión es necesario indicar tanto la dirección IP del servidor como el puerto que estará esperando aceptar las conexiones. La conexión se crea de la siguiente forma:

```
Socket socket=new Socket(
InetAddress.getByName("192.168.43.109"),4000);
```

La dirección IP 192.168.43.109 es la dirección que tendrá el servidor para realizar las pruebas y el puerto 4000 es el que usará el servidor para aceptar conexiones entrantes de los clientes Este nuevo socket generado se usará tanto para la recepción como para el envío de mensajes con el servidor.

5.1.2 Envío de mensajes al servidor

Durante la ejecución de la aplicación, se deberá enviar mensajes que contengan distintos tipos de datos (enteros, cadenas de caracteres…) y cada uno de estos tipo se envía de una forma específica para que la posterior recepción del servidor se realice de forma correcta. Para realizar el envío se necesitará un objeto del tipo `DataOutputStream` el cual se construye del siguiente modo:

```
OutputStream out = Inicializacion.socket.getOutputStream();
DataOutputStream salidaAServer = new DataOutputStream(out);
```

Como puede observarse, para la generación de este objeto se hace uso del socket que se ha creado anteriormente. Una vez tengamos creado el objeto `DataOutputStream` ya podemos realizar envío de mensajes con el servidor. El envío de un número entero desde la aplicación cliente al servidor se realiza de la siguiente forma (`Write.int`):

```
salidaAServer.writeInt (int valor);
```

El otro tipo de dato que se envía durante el proceso de ejecución de la aplicación cliente es una cadena de caracteres. Este envío se realiza por ejemplo cuando se envía la información al insertar un curriculum o al generar un trabajo. En este envío se utiliza la función `WriteBytes` de la siguiente manera:

```
salidaAServer.writeBytes(String información);
```

5.1.3 Recepción de mensajes del servidor

Para poder recibir mensajes del servidor se crea un objeto del tipo `DataInputStream` de la siguiente forma:

```
DataInputStream istream =
new DataInputStream(Inicializacion.socket.getInputStream());
```

Al igual que ocurría en el caso del envío de los mensajes, se utilizará el socket creado en la actividad de inicialización para generar el objeto que permite la recepción. Encontramos distintos tipos de datos que la aplicación cliente debe recibir por parte del servidor, teniendo que tratar cada uno de ellos de forma adecuada. Cuando la recepción es un número entero se realiza de la siguiente forma (método `readInt()`):

```
int valor = istream.readInt();
```

Cuando lo que se obtiene del servidor es una cadena de caracteres se utiliza la función `read()`. Esta función recibe como parámetro un array que será quien finalmente contenga la información enviada por el servidor. La recepción de una cadena de caracteres se realiza de la siguiente forma:

```
byte [] leidos = new byte[200];
istream.read(leidos, 0, leidos.length);
```

5.1.4 Obtener la localización geográfica del dispositivo

Que el dispositivo pueda obtener las coordenadas geográficas de donde se encuentra es un punto fundamental para que el sistema que se propone en este Trabajo Fin de Grado pueda funcionar correctamente. La localización en el sistema juega un papel clave ya que permite realizar una posterior selección correcta de los candidatos potenciales de un determinado trabajo.

Se ha implementado que la forma de obtener la posición geográfica del dispositivo se haga únicamente por Internet y no utilizando el GPS. Esta decisión se ha tomado en base al ahorro de energía para el dispositivo que esto supone y a la no necesidad de obtener una posición tan exacta como la que nos indicaría un GPS. Para obtener la localización en Android en primer lugar se crean dos objetos de tipo `LocationManager` y `Location` de la siguiente forma:

```
locManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);

Location loc =
locManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
```

A continuación se implementa un `listener` que permita actualizar la posición del dispositivo. Este `listener` tratará varias circunstancias posibles que puedan ocurrir durante la ejecución de la funcionalidad como pueda ser que la ubicación del dispositivo cambie, que Internet no está activo en el dispositivo (no permitiendo obtener la ubicación), o que Internet ha sido activado. La implementación de este `listener` es la siguiente:

```
LocationListener locListener = new LocationListener() {
      public void onLocationChanged(Location location) {
            if(trabajando==false){
                  asignarLocalizacion(location);
                  trabajando = true;
            }
      }
public void onProviderDisabled(String provider){}
public void onProviderEnabled(String provider){ }
public void onStatusChanged(String provider, int status, Bundle
extras){}

};
```

Finalmente se registra la ubicación en el objeto `locationManager` creado anteriormente:

```
locManager.requestLocationUpdates(
LocationManager.NETWORK_PROVIDER, 30000, 0, locListener);
```

Una vez tenemos la información de la ubicación en el objeto `Location` podemos obtener la longitud y la latitud de la siguiente forma:

```
Location localización;
ObtenerLocalizacion(localización);
double longitud = localizacion.getLongitude();
double longitud = localizacion.getLatitude();
```

5.1.5 Hash de la contraseña

Para mantener la seguridad de la contraseña que el usuario introduce en el proceso de registro se le aplicará un hash en la propia aplicación cliente

permitiendo así que la interceptación de un mensaje de registro por un usuario no autorizado no sea vulnerable para la integridad y confidencialidad de la propia contraseña. Esta operación hash que se implementa es el `Sha1`, y se realiza de la siguiente forma:

```
public static String sha1(String input) throws
NoSuchAlgorithmException {
MessageDigest mDigest = MessageDigest.getInstance("SHA1");
byte[] result = mDigest.digest(input.getBytes());
StringBuffer sb = new StringBuffer();
for (int i = 0; i < result.length; i++) {
      sb.append(Integer.toString((result[i] & 0xff) + 0x100,
      16).substring(1));
}
return sb.toString();
}
```

5.1.6 Google Maps

La aplicación cliente hace uso de Google Maps para que el usuario introduzca su posición en el proceso de insertado de un curriculum como se muestra en la Figura 37.

Figura 37. Inserción de la ubicación en el proceso de curriculum

Para que la aplicación cliente pueda hacer uso de Google Maps y de toda la funcionalidad que aporta la herramienta, es necesario en primer lugar obtener la "API KEY" de Google Maps[19]. Esta "API KEY" se puede obtener desde la página de google para desarrolladores utilizando para ello la huella electrónica de la aplicación. Una vez se obtiene la clave de google se debe insertar en el `Manifest` de la aplicación para que el uso de la herramienta de Google Maps se pueda realizar:

```
<meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyCGXD20X-KtqHOCFbTTlWwGLqlhCugVWG0"
/>
```

Una vez se tienen los permisos por parte de google para poder hacer uso de la herramienta de Google Maps, la inserción del mapa en la interfaz de la aplicación se realiza de la siguiente forma:

```
<fragment
            android:id="@+id/map"
            android:layout_width="310dp"
            android:layout_height="258dp"
            android:layout_gravity="center"
     android:layout_marginTop="20dp"
            class="com.google.android.gms.maps.SupportMapFragment" />
```

A la hora de manipular el mapa y realizar marcas en él, se implementa el siguiente código:

```
mapa.addMarker(new MarkerOptions()
          .position(Location)
          .title("Tu direccion")
          .snippet("Ubicación actual del usuairo")
          .icon(BitmapDescriptorFactory
                 .fromResource(R.drawable.ic_launcher))
          .anchor(0.5f, 0.5f));
```

5.1.7 Permisos

La aplicación cliente necesita ciertos permisos que el usuario debe aceptar al instalar la aplicación en el dispositivo. Estos permisos deben ser aceptados para que el sistema pueda obtener la ubicación, hacer uso de Google Maps o comprobar el estado de la conexión a Internet para que algunas funcionalidades se realicen o no en función de esto. Los permisos que se deben aceptar al instalar la aplicación cliente son los siguientes:

```
<uses-permission
android:name="org.example.ejemplogooglemaps.permission.MAPS_RECEIVE
" />
    <uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSER
VICES" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
```

5.2 Implementación del servidor

En esta sección se abordará los elementos más importantes de la implementación del servidor. Como se ha explicado anteriormente el servidor esta implementado en lenguaje de programación C y tiene que hacer uso de una base de datos en MySQL.

5.2.1 Aceptar conexión del cliente

Las conexiones de la aplicación cliente con el servidor se aceptan con la siguiente implementación:

```
struct sockaddr_in cliente;
if((socketClient=accept(socketServ,(struct sockaddr *)&cliente,
&size))==-1){
      perror("Cannot accept");
}
```

Como puede observarse, la información referente a la petición de conexión por parte del cliente se almacena en una estructura de tipo `sockaddr_in`. A partir de

este momento en el que se acepta la conexión el cliente puede comenzar con el envío de información y el servidor podrá realizarla.

5.2.2 Envío de mensajes al cliente

El envío de los mensajes se realiza mediante la utilización de la función `send` de la siguiente forma:

```
int valor;
send(socketClient, &valor, sizeof(int), 0);
```

A diferencia de lo que sucedía con la aplicación cliente, se utiliza siempre la función `send` independientemente del tipo de dato que se quiera enviar ya que gracias a que se puede incluir el tamaño de lo que se envía como un parámetro, la recepción en el cliente se realiza de forma correcta.

5.2.3 Recibir mensajes del cliente

Para la recepción de mensajes por parte de la aplicación cliente se hace uso del método `recv` el cual es utilizado de la siguiente forma:

```
int valor;
recv(socketClient, &valor, sizeof(int),0);
```

Al igual que ocurre con la función de envío, no es necesario usar una función distinta a `recv` dependiendo del tipo de variable a recibir. En este ejemplo de recepción de la información del curriculum podemos observar que a pesar de ser una cadena de caracteres lo que se recibe, se sigue usando el mismo método:

```
char curriculum[512];
recv(socketClient, curriculum, sizeof(curriculum),0);
```

5.2.4 Conexión con la base de datos

El servidor se conecta con la base de datos para almacenar la información del usuario y poder gestionarla posteriormente. Debido a la estructura de algunas operaciones que se realizan en el servidor es necesario tener varias conexiones simultáneas a la misma base de datos para poder agilizar las operaciones que puedan paralelizarse (teniendo cuidado de no realizar operaciones de modificación de la base de datos paralelamente con operaciones de consulta de la misma). La conexión se realiza de la siguiente forma:

```
MYSQL *conn;
conn = mysql_init(NULL);
if (!mysql_real_connect(conn, server,user, password, database, 0,
NULL, 0)) {
fprintf(stderr, "%s\n", mysql_error(conn));
            exit(1);
   }
```

5.2.5 Interactuar con la base de datos

Para realizar ejecutar una sentencia en la base de datos desde la propia ejecución del servidor se utiliza la siguiente implementación:

```
if (mysql_query(conn,sentencia_sql)) {
            fprintf(stderr, "%s\n", mysql_error(conn));
       exit(1);
   }
res = mysql_use_result(conn);
while ((row = mysql_fetch_row(res)) != NULL){
//Se recorren las filas que retorna la consulta en el caso de
que lo haga
}
mysql_free_result(res);
```

La variable `sentencia_sql` será una cadena de caracteres con la consulta que se quiere realizar. Si la consulta que se ejecuta no devuelve ningún registro por la naturaleza de la misma, no sería necesario implementar el bucle `while` que observamos en el código.

5.2.6 Cálculo de distancias

El cálculo de las distancias se realiza utilizando la denominada fórmula de Haversine[20]. Esta fórmula permite realizar cálculos de distancias dados dos puntos que pertenecen a una esfera y conociendo únicamente la longitud y latitud de los mismos. La representación de los parámetros de la fórmula puede verse en la Figura 38.

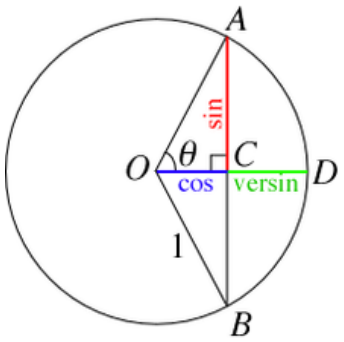

Figura 38. Representación de los parámetros de La Ley de Haversin

Esta fórmula se expresa de forma matemática de la siguiente forma:

$$\operatorname{haversin}\left(\frac{d}{R}\right) = \operatorname{haversin}(\varphi_1 - \varphi_2) + \cos(\varphi_1)\cos(\varphi_2)\ \operatorname{haversin}(\Delta\lambda).$$

Donde:

- Haversin es la función definida como: $(\theta) = \text{sen}^2(\theta/2) = (1-\cos(\theta))/2$
- d es la distancia entre los dos puntos
- R es el radio de la esfera que en nuestro caso pasa a ser el radio de la Tierra
- φ_1 y φ_2 son las latitudes del primer y segundo punto
- $\Delta\lambda$ es el resultado de la resta de longitudes entre los dos puntos

Una posible implementación de esta fórmula en lenguaje C sería la siguiente[21]:

```
#define R 6371
#define TO_RAD (3.1415926536 / 180)

double dist(double th1, double ph1, double th2, double ph2)
{
      double dx, dy, dz;
      ph1 -= ph2;
      ph1 *= TO_RAD, th1 *= TO_RAD, th2 *= TO_RAD;

      dz = sin(th1) - sin(th2);
      dx = cos(ph1) * cos(th1) - cos(th2);
      dy = sin(ph1) * cos(th1);
      return asin(sqrt(dx * dx + dy * dy + dz * dz) / 2) * 2 * R;
}
```

5.2.7 Método para desencadenar

Como se ha mencionado en secciones anteriores, el envío de la información se realiza mediante la concatenación de los valores que se quieran enviar intercalados por el carácter "#". En el servidor se implementa un método que sea capaz de desencadenar los mensajes y se obtengan los elementos de forma eficiente. La implementación de este método se realiza utilizando la funcionalidad `strtok` de la siguiente forma.

```
char * obtener (char * cadena, int pos){
      const char s[] = "#";
      char *token;
      token = strtok(cadena, s);
       int i =0;
       while( token != NULL ) {
            if(i==pos){
                  return token;
            }
            token = strtok(NULL, s);
            i++;
       }
      return NULL";
}
```

Este método recibe como atributos la cadena concatenada y la posición que se quiere obtener de ella. El resultado que retorna es el valor deseado de la cadena concatenada. Por ejemplo, si se tiene la cadena “idUsurio #nombreDeUsuario” y se quiere obtener únicamente el nombre de usuario, llamaría a la función obtener enviando la cadena completa y como segundo atributo el número 1 (se comienza en 0), y se retornaría el nombre de usuario.

6. Evaluación

En este capítulo se describirán las pruebas que se realizan al sistema para comprobar el correcto funcionamiento de éste. Estas pruebas también se realizarán con el fin de analizar la eficiencia y estudiar distintos escenarios que permitan comprobar el comportamiento del sistema.

En primer lugar se realizaran pruebas de rendimiento, encaminadas a evaluar el tiempo de envío de los mensajes. Para estas pruebas es importante la conexión a Internet que tenga el dispositivo móvil y el servidor, por lo que las pruebas se realizarán modificando este parámetro para ver cómo se comportan los resultados. Posteriormente se realiza una prueba en la que se evalúa el algoritmo de selección de los candidatos potenciales a medida que el número de usuarios se incrementa a fin de evaluar la escalabilidad del sistema. Este algoritmo, tiene que recorrer toda la tabla de curriculums de la base de datos, es por esto que es interesante estudiar su comportamiento con distintos tamaños de tabla.

Finalmente se estudiará de qué forma se ha conseguido mejorar la eficiencia del sistema utilizando la concatenación de la información de los mensajes en lugar de enviar una gran cantidad de mensajes que contenga la misma información que el mensaje concatenado pero de manera individual.

6.1 Entorno de evaluación.

Los dispositivos usados para realizar las pruebas tienen una importancia destacable ya que sus características van a influir directamente en los resultados. A continuación se detallan las características de los dispositivos utilizados en las pruebas.

Como dispositivo móvil que contiene la aplicación se ha utilizado un Xiaomi MI3 con un procesador de 2.3 GHz y con 2 GB de memoria RAM. Este dispositivo puede verse en la Figura 39.

Figura 39 Dispositivo Xiaomi MI3

En cuanto al servidor, se ha utilizado un portátil de marca Lenovo U410 el cual posee un procesador a 2,6 GHz y 8GB de memoria RAM. El dispositivo puede observarse en la Figura 40.

Figura 40 Portátil Lenovo U410

6.1 Evaluación tiempos de respuesta

En esta prueba se evaluará como varía el tiempo de envío y recepción de mensajes en función del escenario que se seleccione. Los dos escenarios que se plantean para las pruebas son:

- **Escenario 1**: El servidor y la aplicación cliente se encuentran en una misma red de área local conectadas a un mismo *router*. Esta circunstancia implica que los mensajes no abandonan la red local y no tienen que enviarse a una red externa ya que los elementos necesarios

se encuentran localizados en la misma red. Esta estructura se muestra en la Figura 41.

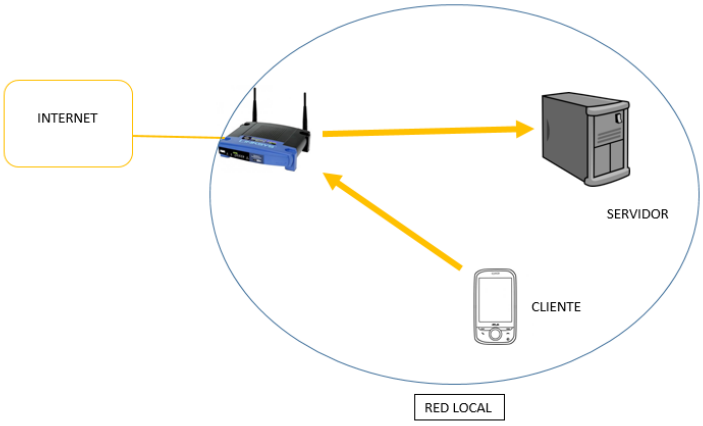

Figura 41. Representación del escenario 1

- **Escenario 2:** El servidor y el cliente se encuentran en distintas redes. Se accede al servidor a través de una dirección IP pública visible y conocida por la aplicación cliente. Para llevar a cabo esta prueba, se usará como máquina servidora el computador guernika.lab.inf.uc3m.es, una máquina perteneciente al departamento de Informática de la Universidad Carlos III cuya dirección IP es 163.117.142.238. En este escenario los mensajes tendrán que salir a Internet para llegar al destino por lo que antes de comenzar las pruebas se presupone que el tiempo de respuesta será mayor. La representación de la red externa puede verse en la Figura 42.

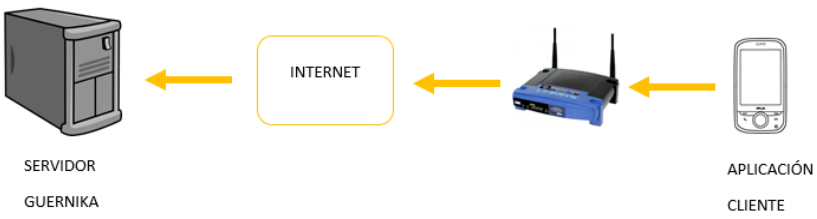

Figura 42. Representación de la red externa

6.1.1 Tiempo de respuesta en el escenario 1 (red local)

Para realizar esta prueba vamos a utilizar el mensaje de petición de conexión que el cliente realiza. El objetivo es evitar que el procesamiento de alguna operación en el servidor provoque un tiempo de respuesta mayor que el que se obtiene únicamente con el envío y recepción del mensaje. Los resultados obtenidos se muestran en la Figura 43 realizando un total de 25 envíos de peticiones de conexión.

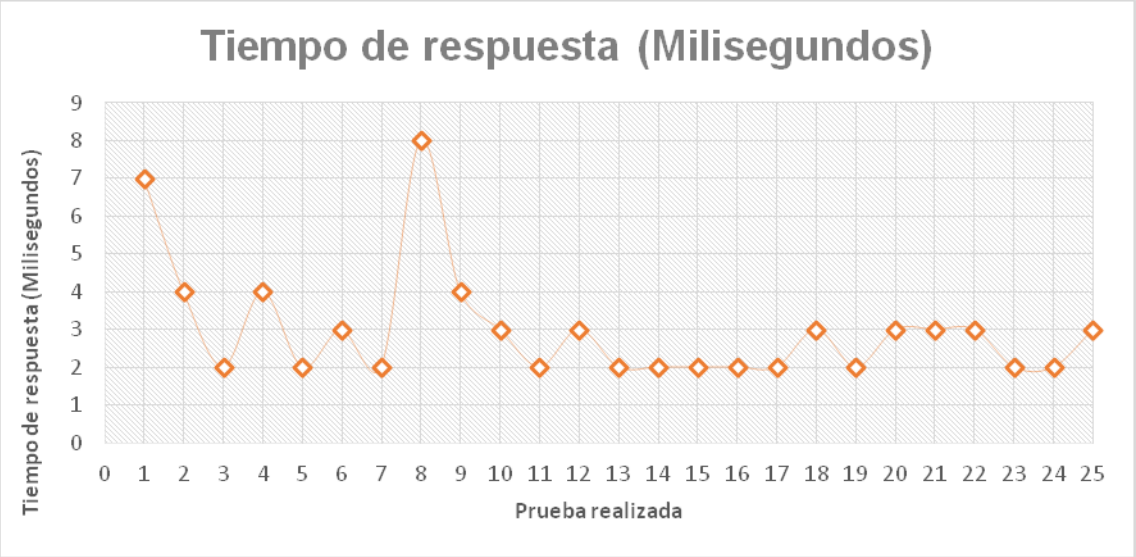

Figura 43. Representación de los tiempos de respuesta en red local

En este escenario nos encontramos que el tiempo de respuesta medio es aproximadamente:

$= 3{,}04$ **milisegundos**

Para analizar el grado de estabilidad del valor de respuesta se calcula la desviación típica de los datos. Este cálculo nos permitirá conocer de qué forma cambian los valores y la desviación de los mismos:

$$S = \sqrt{\frac{\sum_i (Xi - \bar{X})^2}{n}}$$

$$S = 1{,}52\ milisegundos$$

El valor obtenido de la desviación típica es un valor medio teniendo en cuenta los valores de las mediciones. Por lo tanto se puede establecer que el sistema se comporta de una manera estable en el tiempo de respuesta cuando servidor y cliente se encuentran en una misma red local.

6.1.2 Tiempos de respuesta en el escenario 2 (red externa)

Como se ha mencionado anteriormente, se utilizará el servidor de la Universidad guernika.lab.inf.uc3m.es para realizar estas pruebas. Debido a que los mensajes en este caso saldrán a Internet, la conexión del dispositivo a la red es un factor que afecta directamente a la prueba. En las pruebas a realizar se utilizará una red WIFI de 47,97 MB/s de descarga y 5,31 MB/s de subida. Los tiempos de respuesta obtenidos se muestran en la Figura 44.

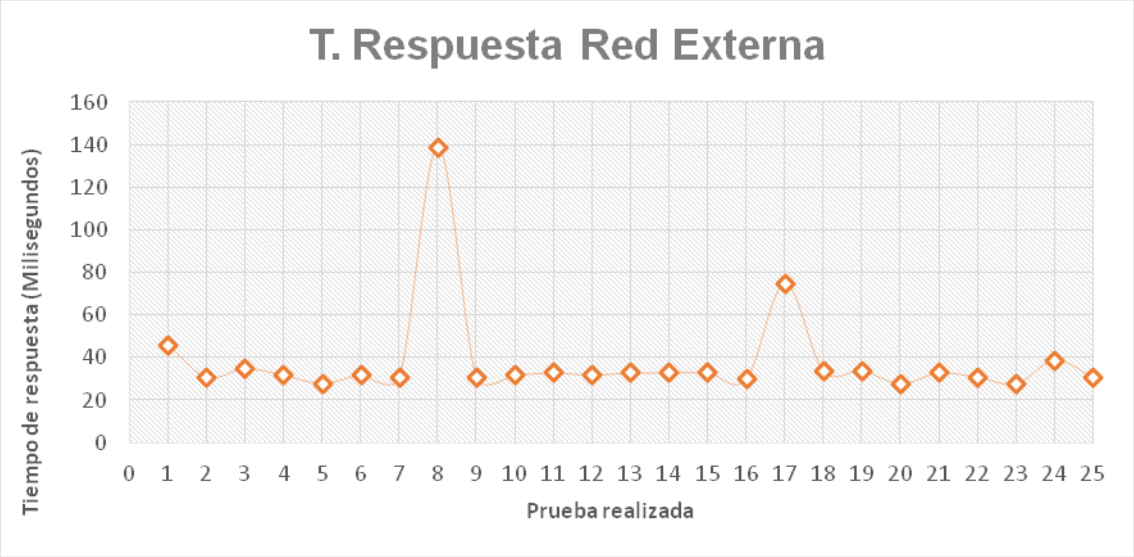

Figura 44. Representación de los tiempos de respuesta en red externa

Como puede observarse en la Figura 44, encontramos dos puntos que difieren mucho del resto debido a su excesivo valor. Esto ha podido producirse por las características de la prueba realizada, es decir, que en este caso nos encontramos con una red externa cuya eficiencia se mide por ciertos parámetros (nivel de

tráfico, número de peticiones). Es posible que en el momento de realizar la prueba la red en ese preciso instante se encontrara más saturada que en el resto de casos, produciéndose esta diferencia de valor. La media que se obtiene de estos datos es:

$\bar{X} = 38{,}56$ **milisegundos**

Por su parte, la desviación típica de los datos muestra el siguiente resultado:

$$S = \sqrt{\frac{\sum_i (Xi - \bar{X})^2}{n}}$$

$$S = 22{,}86 \ milisegundos$$

Analizando los resultados obtenidos, observamos que la desviación típica obtenida es un resultado elevado y por lo tanto la estabilidad de los tiempos de respuesta cuando el servidor y el cliente no se encuentran en la misma red es mucho menor. Este resultado era predecible debido a la configuración interna de la red que no siempre conceden los mismos parámetros a las conexiones. Por ejemplo, un paquete enviado entre A y B podría atravesar 1 o varios routers, encontrar un excesivo tráfico de red en cada router, etc.

6.1.3 Comparación de tiempos de respuesta

Una vez realizada la evaluación utilizando distintos escenarios podemos realizar una comparación entre ambos. En primer lugar encontramos una diferencia importante entre los tiempos de respuesta utilizando la red local y utilizando una red externa. Esta diferencia es entendible como se ha mencionado anteriormente ya que el tiempo de transmisión de la red es diferente. La comparación se puede observar en las Figuras 45 y 46.

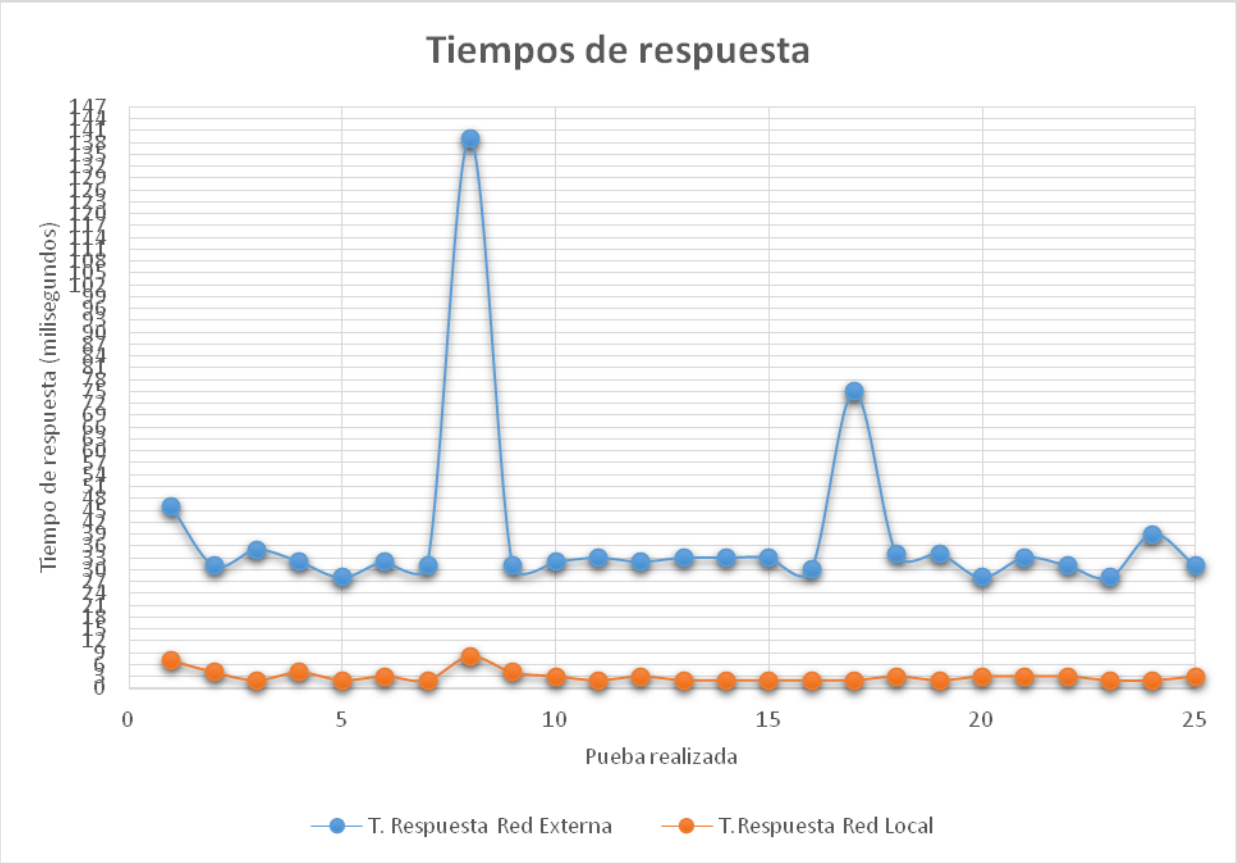

Figura 45. Comparación de los tiempos de respuesta

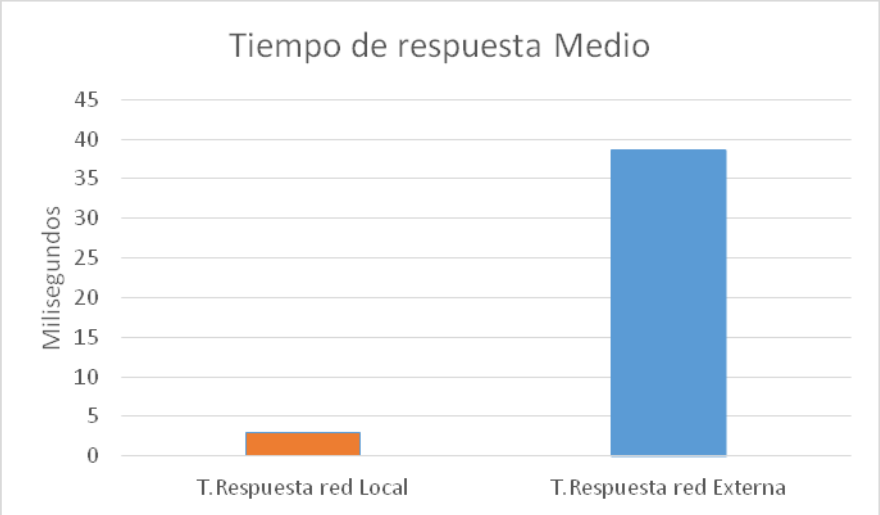

Figura 46. Comparación del tiempo de respuesta medio

En cuanto a la estabilidad de este tiempo de respuesta obtenemos grandes diferencias entre los dos escenarios. El resultado que se obtiene es que utilizando la red local, el tiempo de respuesta es mucha más similar y no existen variaciones significativas entre distintas repeticiones. Sin embargo, la red externa es mucho más variable dando como resultados valores muchos más distantes. La comparación de la variación típica de los dos escenarios se muestra en la Figura 47.

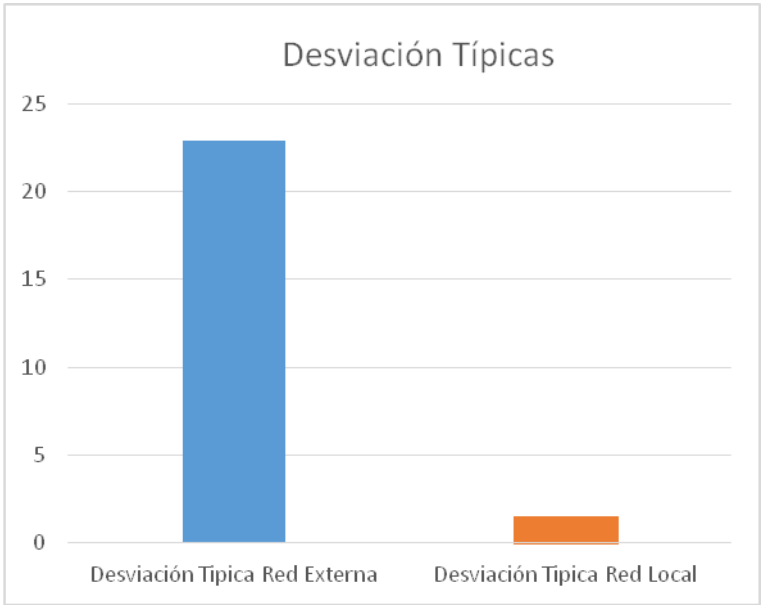

Figura 47. Comparación de las desviaciones típicas

6.2 Evaluación del algoritmo de selección de candidato

El algoritmo de selección de candidatos potenciales se ejecuta cada vez que se genera un nuevo trabajo y tiene como particularidad que tienen que recorrer todos los elementos de la tabla curriculum de la base de datos. Esta circunstancia provoca que a medida que el sistema tenga un número más elevado de curriculums almacenados el tiempo de cómputo del algoritmo será mayor. Es interesante observar cómo crece el tiempo de cómputo del algoritmo a medida que la tabla curriculums crece para realizar estimaciones posteriores que indiquen la eficiencia del algoritmo utilizando la metodología escogida. Las pruebas realizadas para distintos tamaños de la tabla curriculum aportan los resultados que pueden verse en la Figura 48.

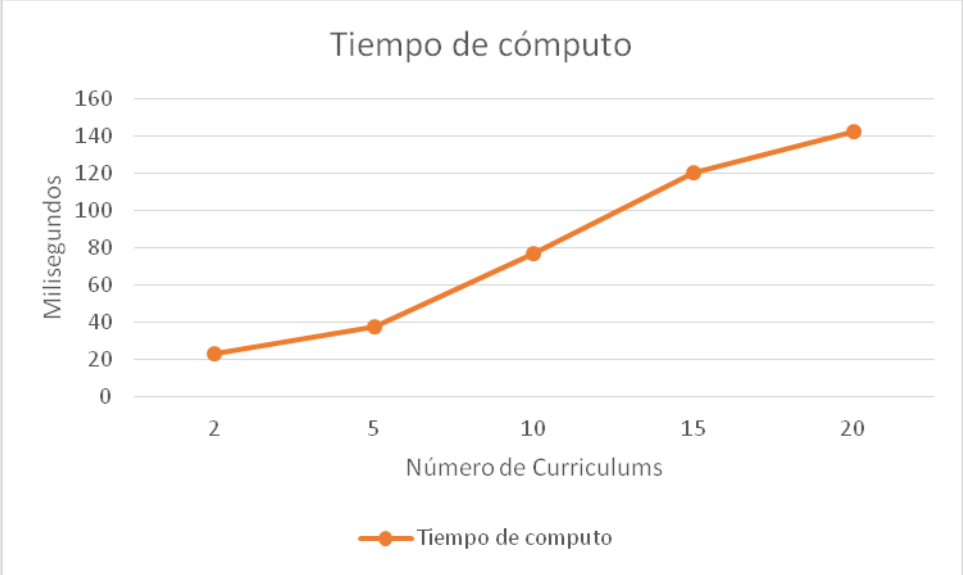

Figura 48. Evolución tiempo de cómputo del algoritmo de selección

Como se puede observar el tiempo de cómputo experimenta una subida importante a medida que se aumenta el número de usuarios del curriculum. En las pruebas realizadas, se pasa de 23 milisegundos de cómputo cuando hay dos curriculums en el sistema a 142 milisegundos cuando hay 20. Este algoritmo afecta de manera activa al tiempo de cómputo global debido a que también se realiza la ejecución de la fórmula de Harversine por cada curriculum.

No es difícil predecir que el número de usuarios que compondrán el sistema será un número tan alto que el actual procedimiento de selección de candidato no podrá llevarse a cabo y tendrán que buscarse alternativas o modificar las características existentes para que el procedimiento se realice de una manera más eficiente.

6.3 Evaluación de la eficiencia al concatenar mensajes

En el proceso de diseño, se tomó la decisión que los mensajes fueran la concatenación de varios elementos de forma que utilizando un único mensaje se podría enviar la misma información que numerosos mensajes de un único elemento. El motivo principal de usar esta técnica es evitar un tráfico excesivo y mejorar el proceso de comunicación entre el servidor y la aplicación cliente. La eficiencia que se consigue utilizando esta concatenación varía según el caso que se

analice. A mayor número de elementos concatenados se obtiene una mayor eficiencia en comparación con el envío individual de los elementos. Tras una serie de pruebas realizadas sobre el envío de un mensaje de registro, se observa que la longitud del mensaje no afecta al tiempo de envío del mismo, y si lo hace es imperceptible para el sistema. Esta circunstancia añade aún más fundamentos sobre la eficiencia de usar la técnica de concatenación de la información.

Para comprobar la eficiencia se utiliza el caso en el que el sistema realiza un mayor envío de información concatenada. Esto se produce en el proceso de enviar la información de un curriculum que el usuario acaba rellenar. La información que se envía en este mensaje es la siguiente:

```
IdUsuario#Nombre#Apellidos#nombreEmpresa#Latitud#
Longitud#Campo#Distancia#Descripción#Teléfono#Email#OtroContacto
```

Se envían un total de 12 elementos concatenados y teniendo en cuenta que el tiempo medio de envío de un mensaje utilizando la red local es de 3 milisegundos se puede comprobar la eficiencia obtenida respecto al envío individual es muy elevada. Si el envío se hubiera hecho por partes enviando elemento a elemento el tiempo de transmisión de los mensajes se puede predecir que hubiera sido:

$$T\,(milisegundos) = NumMensajes * TMedioMensaje$$
$$T\,(milisegundos) = 12 * 3 = 36\ milisegundos$$

Es decir, el envío se produce cada3 milisegundos utilizando la concatenación, mientras que el envío individual de cada dato tardaría 36 milisegundos. La comparación puede verse en la Figura 49.

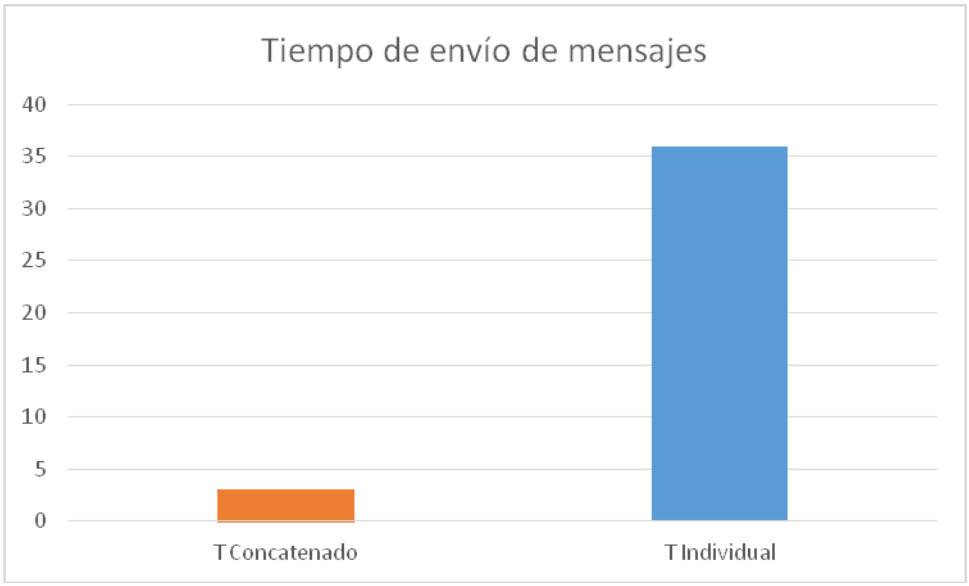

Figura 49. Comparación de tiempo de envío

7. Conclusiones

Este capítulo se centra en la revisión de los objetivos que se establecieron en el primer capítulo, estudiando su situación y cumplimiento en el estado actual del proyecto. Además de esto, se propondrán nuevas líneas de trabajo futuro para que este proyecto pueda ser el comienzo de nuevos desarrollos o investigaciones. Por último se mostrarán un presupuesto del proyecto y una evaluación personal del mismo.

7.1 Revisión de objetivos

Para realizar la revisión de los objetivos, vamos a estudiar y analizar la situación actual del proyecto conforme a los objetivos marcados en el primer capítulo.

En primer lugar, el objetivo general era el desarrollo de un sistema distribuido que permitiera la creación y la gestión de trabajos desde una misma aplicación. El sistema actual permite primeramente que usuarios puedan insertar sus capacidades y datos personales para optar a trabajos permitiendo la inserción de curriculums en el sistema. Así mismo se pueden generar trabajos seleccionando las cualidades de éstos y los candidatos potenciales de los mismos. Toda esta funcionalidad se realiza utilizando un sistema distribuido en el que una aplicación móvil desarrollada en Android actual como cliente y envía peticiones al servidor.

Otro objetivo marcado, fue el de la posibilidad de generar una gran variedad de trabajos en el sistema. Se han añadido una gran cantidad de trabajos posibles y se ha diseñado de tal forma que la incorporación de nuevos tipos de trabajos en el sistema sea una tarea sencilla y que no afecte al resto de la implementación.

En el capítulo 1 también se expuso como objetivo la facilidad que debería suponer para el usuario la utilización del sistema, ya que éste no tenía por qué tener unos conocimientos elevados en el uso de estas herramientas. La aplicación cliente (único elemento del sistema visible para el usuario) ha sido diseñada con una

interfaz altamente intuitiva, permitiendo que el usuario interactúe con el sistema de forma sencilla. Para el diseño de esta interfaz ha sido necesario el uso de patrones de diseño ya establecidos así como elementos de la interfaz que ya son fácilmente reconocibles para el usuario (uso de botones, pestañas…).

Otro objetivo fue la elaboración de un protocolo de comunicación eficiente entre la aplicación cliente y el servidor. En primer lugar en el sistema se establece un código de operación para cada funcionalidad que se permite, permitiendo de este modo realizar comunicaciones mucho más estructuradas y que además darán la posibilidad de añadir nuevas funcionalidades de una manera sencilla. En el momento en el que la aplicación cliente le envía el código de operación el servidor, ambos elementos pueden establecer las estructuras de los mensajes que tendrán que enviar y recibir. Además de esto, la incorporación de la técnica de concatenación de los elementos a enviar ha permitido una gran eficiencia en los tiempos de envío de mensajes como puede observarse en el capítulo de la evaluación.

Que el sistema fuera totalmente móvil para el usuario y que pudiera llevarse consigo fue otro objetivo marcado en el primer capítulo. Gracias al desarrollo de la aplicación en un dispositivo móvil, el sistema se puede ser utilizado en todos los lugares únicamente llevando consigo el dispositivo móvil. Otra característica importante del sistema que favorece la movilidad es que la información del usuario (curriculum, trabajos generados…) es totalmente independiente de dispositivo que se utilice, es decir que la información no se almacena en la herramienta sino que se descarga del servidor. De esta forma el usuario podrá generar trabajos o visualizar las ofertas desde dispositivos que no necesariamente tengan que ser el mismo que usa normalmente.

Finalmente, el último objetivo marcado en el capítulo 1 fue que la aplicación cliente utilizara las máximas funcionalidades del dispositivo móvil para aportar valores positivos al sistema. La aplicación cliente actual utiliza la conexión a Internet del dispositivo para realizar las comunicaciones con el servidor. Además de esto se

utiliza la ubicación que el dispositivo proporciona permitiendo añadir opciones y mejorando el sistema en conjunto.

7.2 Líneas futuras

Durante la realización del proyecto, han surgido algunas posibilidades que pensamos podrían mejorar el sistema y que podrían ser el comienzo del estudio de trabajos futuros. Estas líneas futuras de trabajo se describen a continuación:

- En primer lugar se debería mejorar la seguridad del sistema realizando una autenticación de todos los mensajes. Sería interesante que cuando una aplicación cliente por ejemplo recibe un mensaje del servidor se pueda garantizar que éste sea quien dice ser (autenticación).
- Además, sería interesante el desarrollo de la aplicación cliente usando otros sistemas operativos móviles. El desarrollo del mismo sistema para dispositivos IOS por ejemplo permitiría aumentar considerablemente los usuarios potenciales del sistema y en consecuencia ofertar una mayor cantidad de trabajos.
- La escalabilidad del sistema sería un objetivo futuro muy a tener en cuenta. Después de lo evaluado anteriormente, observamos que el algoritmo de selección de candidato necesita ser más eficiente para evitar pérdidas importantes de tiempo de ejecución cuando el número de usuarios que tenga que analizar aumente considerablemente. Por lo tanto una línea de futuro que se podría mejorar del sistema es el proceso de selección del candidato.

7.3 Presupuesto

En la sección de presupuestos se realizará un análisis de costes para llevar a cabo la realización del proyecto. Estos costes serán divididos en recursos humanos y recursos materiales utilizados.

7.3.1 Recursos humanos

Los recursos humanos que se han utilizado durante el desarrollo del sistema se analizarán haciendo uso de las distintas fases de elaboración de la que consta el proyecto. Las fases de elaboración del software son:

- Obtención de los conocimientos sobre el estado de la cuestión y análisis del entorno del proyecto.
- Análisis.
- Diseño.
- Implementación.
- Evaluación y pruebas.
- Documentación.

Hay que tener en cuenta que para el desarrollo del sistema se ha utilizado un único recurso humano (una única persona). En cuanto al tiempo total de desarrollo del proyecto, se establece el comienzo de éste el 1 de Diciembre del 2013 y su finalización el 22 de Junio de 2014 (es decir, un total de 203 días).
Realizando una estimación donde el trabajador aproximadamente cobre unos 40 €/hora y trabaje 30 horas semanales, se establecen un total de 1160 horas dedicadas teniendo en cuenta que el periodo de trabajo han sido 29 semanas. En cuanto a la repartición del tiempo y el coste asociado resultan unos datos como el que se muestra en la tabla 37.

	Precio / Hora	**Horas**	**Coste de la fase (€)**
Análisis del problema	30	232	6.969
Diseño	30	174	5.220
Implementación	30	348	10.440
Evaluación y pruebas	30	116	3.480
Documentación	30	290	8.700
Total	30	1160	**34.809**

Tabla 39 Coste recursos humanos

7.3.2 Recursos materiales

En los costes de los materiales se tendrán en cuenta todos los elementos hardware utilizados así como los servicios que se han tenido que utilizar durante el desarrollo. En la tabla 40 se muestra este coste.

	Cantidad	Coste(€)	Coste total (€)
Ordenador portátil	1	690	690
Conexión a Internet (mensual)	7	45	315
Dispositivo móvil	1	400	400
Impresión del proyecto	3	35	105
Total			**1.510**

Tabla 40 Costes materiales

7.3.3 Costes totales

Los costes totales del proyecto se establecen como la suma de recursos humanos y los recursos materiales. Este coste puede verse en la tabla 41.

	Coste total (€)
Recursos humanos	34.809
Conexión a Internet (mensual)	1.510
Total	**36.319**

Tabla 41 Costes totales

7.3.4 Plantilla resumen

A continuación se muestra una plantilla que resume todos los datos del presupuesto:

UNIVERSIDAD CARLOS III DE MADRID
Escuela Politécnica Superior

PRESUPUESTO DE PROYECTO

1.- Autor: Iván Álvarez Arias

2.- Departamento: Departamento de informá

3.- Descripción del Proyecto: Desarrollo de una aplicación móvil Android para la gestión de empleo

- Título	**Desarrollo de una aplicación móvil Android para la gestión de empleo**
- Duración (meses)	**7 Meses**
Tasa de costes Indirectos:	20%

4.- Presupuesto total del Proyecto (valores en Euros):

Euros

5.- Desglose presupuestario (costes directos)

PERSONAL

Apellidos y nombre	N.I.F. (no rellenar - solo a titulo informativo)	Categoría	Dedicación (hombres mes) [a]	Coste hombre mes	Coste (Euro)	Firma de conformidad
Iván Álvarez Arias		Analista	1,76	3.959,65	6.968,98	
Ivan Álvarez Arias		Diseñador	1,32	3.954,54	5.219,99	
Iván Álvarez Arias		Ingeniero	2,65	3.939,62	10.439,99	
Ivan Álvarez Arias		Documentación	2,21	3.936,65	8.700,00	
					0,00	
		Hombres mes	**7,94**	**Total**	31.328,97	

[a] 1 Hombre mes = 131,25 horas. Máximo anual de dedicación de 12 hombres mes (1575 horas)
Máximo anual para PDI de la Universidad Carlos III de Madrid de 8,8 hombres mes (1.155 horas)

EQUIPOS

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable [d]
Dispositivo Móvil	300,00	40	7	60	14,00
Portatil	650,00	90	7	60	68,25
					0,00
					0,00
					0,00
					0,00
				Total	82,25

[d] Fórmula de cálculo de la Amortización:

$$\frac{A}{B} x C x D$$

A = nº de meses desde la fecha de facturación en que el equipo es utilizado
B = periodo de depreciación (60 meses)
C = coste del equipo (sin IVA)
D = % del uso que se dedica al proyecto (habitualmente 100%)

SUBCONTRATACIÓN DE TAREAS

Descripción	Empresa	Coste imputable
Impresión del documento	Copisteria	105,00
	Total	105,00

OTROS COSTES DIRECTOS DEL PROYECTO[e]

Descripción	Empresa	Costes imputable
Conexión a internet	ONO	315,00
	Total	315,00

[e] Este capítulo de gastos incluye todos los gastos no contemplados en los conceptos anteriores, por ejemplo: fungible, viajes y dietas,

6.- Resumen de costes

Presupuesto Costes Totales	**Presupuesto Costes Totales**
Personal	31.329
Amortización	82
Subcontratación de tareas	105
Costes de funcionamiento	315
Costes Indirectos	6.366
Total	38.197

7.4 Evaluación personal

Estando cada vez más cerca del final del periodo académico en la Universidad me siento afortunado de las capacidades que he adquirido en ella. La realización de un proyecto como este sirve para darse cuenta de lo que uno es capaz de hacer aplicando únicamente los conocimientos aprendidos durante estos años.

La satisfacción personal con el proyecto es elevada, ya que se ha conseguido un sistema que perfectamente puede incorporase a la sociedad y que en mi opinión prestaría un servicio interesante para gran parte de ella. El mero hecho de que una persona pueda obtener algún trabajo con el uso de este sistema sería de una satisfacción enorme, poder comprobar que tu trabajo ayuda a la gente.

La elección de este proyecto se eligió como forma de demostrar que las capacidades adquiridas durante estos años pueden utilizase perfectamente para que la sociedad avance, para solucionar los problemas que nos encontramos en ella. La búsqueda de alternativas y nuevas ideas tiene que ser la tónica general de nuestras vidas ahora, como futuros ingenieros.

Referencias

[1] Dalvik - http://source.android.com/devices/tech/dalvik/

[2] IOS - http://www.apple.com/es/ios/

[3] BlackBerry OS - http://es.blackberry.com/

[4] Windows - http://windows.microsoft.com/es-es/windows/home.

[5] Lenguaje de programación C -
http://www.openstd.org/JTC1/SC22/WG14/www/standards

[6] MySQL - http://www.mysql.com/

[7] Laboris empeo - http://www.laboris.net/

[8] Cb trabajo y empleo - http://www.cbgroupinternational.com/

[9] Trovit - http://www.trovit.es/

[10] Uber - https://www.uber.com/

[11] TCP - http://www.networksorcery.com/enp/protocol/tcp.htm

[12] UDP - http://www.erg.abdn.ac.uk/~gorry/eg3561/inet-pages/udp.html

[13] Google Maps - https://developers.google.com/?hl=es

[14] Ley de Haversine - http://rosettacode.org/wiki/Haversine_formula

[15] http://www.descargarandroid.com/la-historia-de-android/

[16]https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2arquitectura-de-android

[17] http://developer.android.com/about/index.html

[18] http://www.ecured.cu/index.php/Historia_del_Lenguaje_C

[19] http://alt1040.com/2011/10/historia-de-la-tecnologia-lenguaje-c

[20] http://www.slideshare.net/anderson_rodriguez/historia-de-mysql

[21] http://rosettacode.org/wiki/Haversine_formula

Anexo. Manual de Usuario de la Aplicación

En este anexo se mostrará un manual de usuario que permita utilizar la aplicación cliente de forma correcta.

En primer lugar para abrir la aplicación se deberá pulsar sobre el icono creado en el dispositivo después de realizar el proceso de instalación:

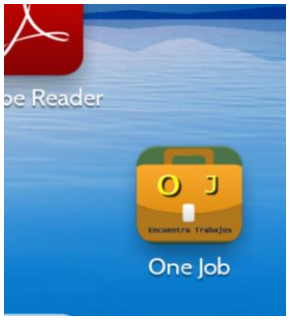

Figura 50. Icono en el dispositivo

Después de pulsar el logotipo la aplicación se abrirá mostrando la siguiente pantalla de Login:

Figura 51. Pantalla de Login

Como puede observarse, se permite introducir los datos del nombre del usuario y la contraseña. Una vez introducidos estos datos se pulsará sobre el botón entrar

para autenticarse en el sistema. En el caso de que el usuario no se encuentre aún en el sistema deberá pulsar sobre el texto en color azul que le indica que le registre, una vez presionado ese botón aparecerá la siguiente pantalla:

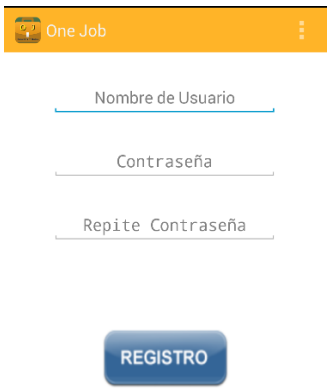

Figura 52. Pantalla de registro

El usuario deberá introducir un nombre usuario nuevo que ya no se encuentre en el sistema y la contraseña dos veces. Después de introducir esta información se pulsará el botón de registro para dar por finalizado este proceso. Si todo el proceso de registro ha salido satisfactoriamente se mostrará la siguiente pantalla:

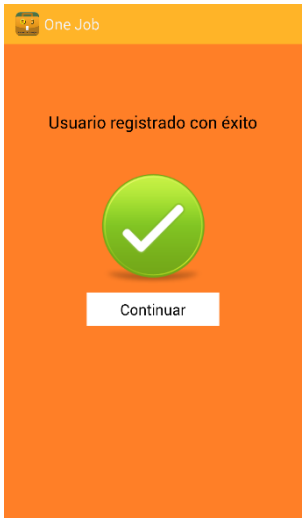

Figura 53. Pantalla de verificación de registro

Una vez el usuario se haya registrado correctamente, ya puede introducir sus nuevos datos en la pantalla de Login vista anteriormente.

Si el Login es correcto se mostrará la pantalla principal:

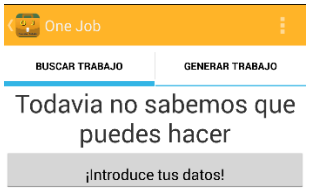

Figura 54. Pantalla principal

Esta pantalla principal como se puede observar consta de dos pestañas. En la primera de ellas se gestionará todo lo relacionado con buscar trabajo (introducir curriculum, visualizar ofertas) y en la segunda se gestionará la generación de

trabajo. Si se pulsa sobre el botón que se observa para introducir tus datos, nos encontramos con la inserción de curriculum la cual consta de los siguientes pasos:

1. Datos personales:

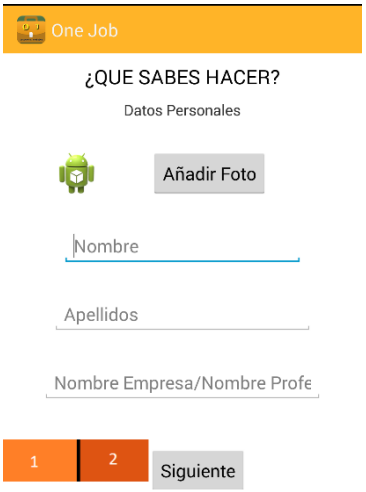

Figura 55. Pantalla curriculum 1

2. Ubicación del usuario:

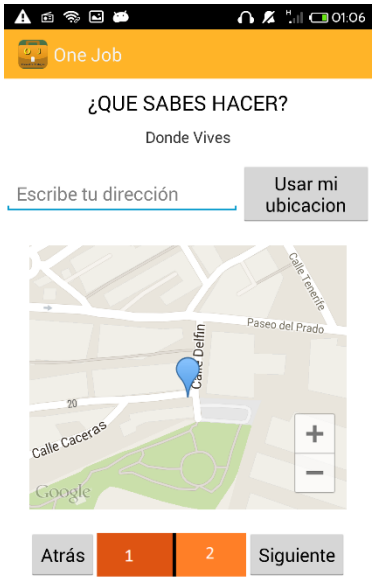

Figura 56. Pantalla curriculum 2

3. Datos de formación y características:

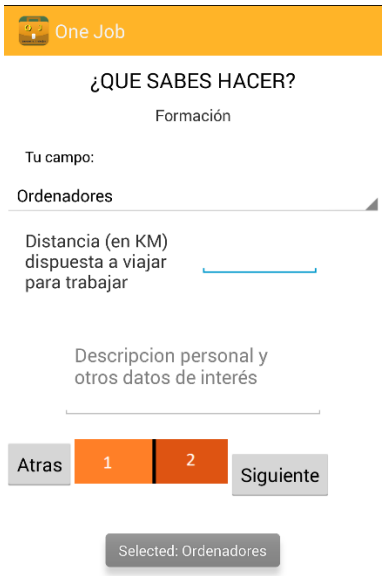

Figura 57. Pantalla curriculum 3

4. Formas de contacto:

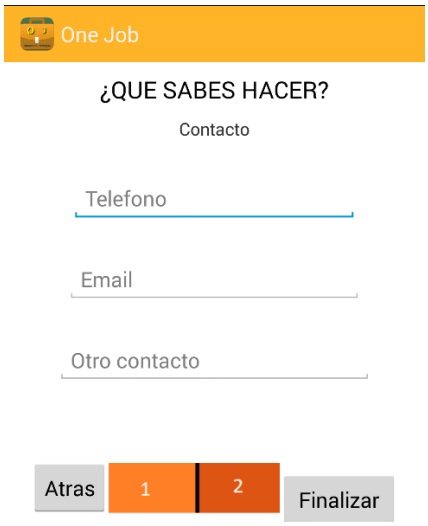

Figura 58. Pantalla curriculum 4

En la segunda pestaña dela aplicación principal se pueden generar los trabajos y listar los ya generados:

Figura 59. Pantalla principal 2

Pulsando sobre el botón que permite generar el trabajo se muestra los elementos que son necesarios introducir para generar uno nuevo. Esta pantalla es la siguiente:

Figura 60. Pantalla de generar trabajo